

PRIORITY DOCUMENT
SUBMITTED OR TRANSMITTED IN
COMPLIANCE WITH
RULE 17.1(a) OR (b)



EPO - Munich
83

12 Sep. 2003

REC'D 02 OCT 2003

WIPO PCT

**Prioritätsbescheinigung über die Einreichung
einer Patentanmeldung**

Aktenzeichen: 102 34 634.8

Anmeldetag: 29. Juli 2002


Anmelder/Inhaber: Baumüller Anlagen-Systemtechnik GmbH &
Co, Nürnberg/DE

Bezeichnung: Fehlerdiagnosesystem mit synchronisierter
Vernetzung und asynchroner Kopplung an
die Leitebene

IPC: G 05 B 23/02

Die angehefteten Stücke sind eine richtige und genaue Wiedergabe der ursprünglichen Unterlagen dieser Patentanmeldung.

München, den 27. August 2003
Deutsches Patent- und Markenamt
Der Präsident
Im Auftrag


Leitang



Spezifikation Baudis NET

Stand: 17.07.2002

T. Tschaftary, S. Büchner, C. Dederding
H. Meis

Inhaltsverzeichnis

1 Überblick	4
2 Definition wichtiger Begriffe	5
3 Anforderungsanalyse	6
3.1 Allgemeine Anforderungen aus der Sicht des Endkunden.....	6
3.2 Anforderungen des Hauptkunden (MAN)	7
3.3 Anforderungen der Fa. Baumüller	7
3.4 Einschränkungen von Baudis und Baudis pro	8
3.5 Schlussfolgerungen	9
4 Gesamtkonzept BAUDIS NET	10
4.1 Spezifikation wichtiger Eigenschaften von BAUDIS NET	10
4.2 Überblick Grobstruktur BAUDIS NET	12
4.3 Kundennutzen	15
5 Spezifikation der Bedienoberflächen	16
5.1 Bedienoberfläche für die Anlageninbetriebnahme, – überwachung und Diagnose 16	
5.2 Bedienoberfläche zur Anlagenkonfiguration	32
6 Spezifikation des Application-Servers	34
6.1 Systemüberblick	34
6.2 Die Serverkomponenten	36
6.3 Objektorientierter Entwurf des Application Servers.....	39

Änderungsgeschichte

Version	Datum	Ersteller	Art der Änderung	Freigabe
1.0	17.07.02	Tschafary, Meis	Ersterstellung	

1 Überblick

Die vorliegende Spezifikation spezifiziert das Softwarewerkzeug BAUDIS NET.

BAUDIS NET soll die gesamte Funktionalität, die zur Konfiguration, Bedienung, Diagnose und Inbetriebnahme von einer großen Anzahl von Antrieben notwendig ist, umfassen.

Um die wesentlichen Anforderungen zu definieren, die bei der Entwicklung von BAUDIS NET berücksichtigt werden sollen, wird im Kapitel 3 eine umfassende Analyse der Anforderungen vorgenommen. Hierbei werden die Anforderungen an BAUDIS NET aus der Sicht eines Endkunden, des Hauptkunden MAN und der Sicht der Fa. Baumüller analysiert. Zusammen mit den Einschränkungen, welchen die bestehenden Softwarewerkzeuge Baudis und Baudis pro unterliegen, können wesentliche Eckpunkte der Softwarestruktur spezifiziert werden.

Aufbauend auf der Anforderungsanalyse in Kapitel 3 wird in Kapitel 4 das Gesamtkonzept für BAUDIS NET spezifiziert. Grundlegend ist eine komponentenbasierte und erweiterbare Infrastruktur die mit Hilfe von flexiblen Funktionen zur Diagnose und Inbetriebnahmeunterstützung erweitert werden kann. Zum Abschluss des Kapitel 4 werden wichtige Vorteile der Softwarestruktur von BAUDIS NET aus der Sicht des Kunden aufgelistet.

Das Kapitel 5 beinhaltet die technische Spezifikation der Bedienoberflächen. Mit Hilfe der Unified Modeling Language (UML) werden die für die in den vorausgehenden Kapiteln notwendigen Funktionen detailliert spezifiziert. Auf der Grundlage dieser Spezifikation wurde ein Oberflächenprototyp in Microsoft Powerpoint erstellt. Er soll zur Optimierung der Bedienoberfläche, als Designstudie sowie als Diskussionsgrundlage für die Einführung weiterer Funktionen dienen.

Das Kapitel 6 spezifiziert die Architektur des BAUDIS NET Application Servers. Hierfür werden die in Kapitel 5 spezifizierten Funktionen mit Hilfe von UML beschrieben und ein objektorientierter Entwurf erstellt.

Bemerkung:

Die vorliegende Spezifikation wird bis zum Abschluss der Spezifikationsphase ständig überarbeitet und erweitert. Sie stellt den Wissensstand und die Planungen zum Zeitpunkt der Erstellung dar. Die Aussagen in den Kapiteln 3 und 4 (Anforderungsanalyse und Gesamtkonzept) sind als gefestigt zu betrachten. Diese werden für die Genehmigung des Projekts als wesentlich angesehen. Die Aussagen in den Kapiteln 5 und 6 (Bedienoberflächen und Application Server) können noch verändert und ergänzt werden.

2 Definition wichtiger Begriffe

Ereignis	Ein Ereignis ist eine Information die von einem Antrieb beim Auftreten an den BAUDIS NET-Server gesendet wird. Es erscheint in der Ereignisanzeige der Bedienoberfläche und im Logbuch. Ereignisse sind z.B. Fehlermeldungen, Meldungen über den Start/ Stop von Aufzeichnungen, Wartungsmeldungen etc. Jedes Ereignis hat eine eindeutige Ereigniskennung über die eine Ereignisbeschreibung in der Dokumentation abgerufen werden kann.
Aufzeichnung	Mit einer Aufzeichnung kann man beliebige Parameterverläufe von beliebigen Reglern erfassen und in einer Datenbank speichern.
Überwachungsansicht	Die Überwachungsansicht ist eine graphische Darstellung eines oder mehrerer Parameter von einem oder mehrerer Regler. Sie dient dazu, den Werteverlauf dieser Parameter hinsichtlich Abweichungen von der Norm zu überwachen. (z.B. Überwachung der Motortemperatur)
Parameterliste	Die Parameterliste enthält alle an einem Reglertyp vorhandenen Parameter
Parametergruppe	Eine Parametergruppe fasst mehrere Parameterseiten zusammen und benennt diese mit einem Namen und einem Kommentar
Parameterseite	Eine Parameterseite ist Teil einer Parametergruppe und enthält mehrere Parameter von beliebigen Reglern.
Langzeitaufzeichnung	Eine Aufzeichnung, deren Daten auf dem BAUDIS NET Server in einer Datenbank gespeichert werden. Gegenteil zur Ringspeicheraufzeichnung
Ringspeicheraufzeichnung	Eine Aufzeichnung, deren Daten im Ringspeicher des G-Drives gespeichert werden. Erst nach Beenden der Aufzeichnung können die Daten auf dem BAUDIS NET Server gespeichert werden.
Konfigurations-Wizard	Eine Abfolge von einzelnen Seiten auf denen der Benutzer Einstellungen vornehmen kann. Jeder Schritt in der Konfiguration umfasst eine Anzahl von Funktionen und wird auf einer Seite dargestellt. Je nachdem wie der Benutzer sich im vorherigen Schritt verhält, wird eine entsprechende Nachfolgeseite angezeigt. (z.B. bei Konfiguration der Aufzeichnung: Auswahlmöglichkeit im Schritt 1: Ringspeicher oder Langzeitaufzeichnung; Je nach Auswahl bekommt der Anwender Seiten für die Konfiguration des Ringspeichers oder der Langzeitaufzeichnung angezeigt.)

3 Anforderungsanalyse

An die Entwicklung einer modernen Software werden hinsichtlich Struktur und Funktion zahlreiche Anforderungen gestellt. Voraussetzung für eine zukunftssichere Softwarearchitektur ist eine sorgfältige Analyse der spezifischen Anforderungen und Funktionen, die die zu entwickelnde Software erfüllen soll. Die folgenden Kapitel geben eine Übersicht, über die an die Entwicklung von BAUDIS NET gestellten Anforderungen. Zunächst werden in Kapitel 3.1 allgemeine Anforderungen aus der Sicht des Endkunden betrachtet. Diese werden präzisiert durch Kapitel 3.2, welches einen Auszug aus einer „Wunschliste“ des Hauptkunden MAN enthält. Zusammen mit den Anforderungen der Fa. Baumüller in Kapitel 3.3 ergibt sich ein umfassendes Anforderungsprofil für die Entwicklung von BAUDIS NET. Die weitere Vorgehensweise bei der Spezifikation wird in Kapitel 3.5 erläutert.

3.1 Allgemeine Anforderungen aus der Sicht des Endkunden

Ziel der Entwicklung von BAUDIS NET ist es, ein Softwarewerkzeug zu entwickeln, das auf die Anforderungen und Bedürfnisse des Endkunden zugeschnitten ist. Aus der Sicht des Endkunden ergab die Analyse folgenden Anforderungen:

(1) Flexible Funktionen zur Überwachung und Diagnose großer Antriebssysteme

Ziel des Projekts „BAUDIS NET“ ist die Entwicklung eines Softwarewerkzeugs zur Überwachung und Diagnose von großen Antriebssystemen. BAUDIS NET soll umfangreiche, flexible Funktionen bieten, die auf die z.T. sehr unterschiedlichen Bedürfnisse der verschiedenen Benutzergruppen beim Kunden zugeschnitten sind.

(2) Einfach zu bedienende, übersichtliche Bedienoberfläche

Die Bedienoberfläche von BAUDIS NET ist der einzige Teil der Software, mit der der Kunde in Kontakt kommt. Insofern ist sie das „Aushängeschild“ der Software und für deren Akzeptanz und Beurteilung von Seiten des Kunden entscheidend.

Beim Entwurf der Bedienoberfläche ist darauf zu achten, dass sie auch für wenig geschultes Personal leicht zu bedienen ist. Die Vielzahl der bei der Diagnose von großen Anlagen anzuzeigenden Informationen müssen graphisch aufbereitet werden und in einer ergonomischen Darstellung dem Anwender präsentiert werden.

(3) Verkürzung der zur Auffindung eines eventuellen Fehlers notwendigen Zeit

Der Nutzen von BAUDIS NET soll für den Kunden darin liegen, dass er sofort nach dem Auftreten eines Fehlers im Antriebssystem die für die Auffindung und Behebung des Fehlers notwendigen Informationen präsentiert bekommt. Hierdurch kann die Zeit in der die Anlage nicht produktiv ist, reduziert werden.

(4) Situationsabhängige Darstellung von Diagnose-Informationen

BAUDIS NET soll die richtige Information zum richtigen Zeitpunkt am richtigen Ort zur Verfügung stellen:

- Aufbereitung der Diagnose-Information gemäß den Anforderungen des jeweiligen Benutzerkreises (z.B. Anlagenbediener, Techniker, Inbetriebnehmer) (→ die richtige Information)
- Zeitnahe Anzeige von Diagnose-Informationen direkt nach Auftreten eines Fehlers (→ zum richtigen Zeitpunkt)
- Zugriff auf BAUDIS NET von beliebigen PCs des Kunden ohne Installationsaufwand – sowohl im lokalen Kundennetz als auch über Fernzugriff. (→ am richtigen Ort)

(5) Reduzierung von kritischen Anlagenzuständen durch vorbeugende Wartung und ständige Überwachung der Anlage

Zukünftig soll BAUDIS NET Mechanismen beinhalten, die dazu beitragen, eventuell bevorstehende Ausfälle von Geräten vorab zu erkennen und dem Kunden mitzuteilen. So kann die Zuverlässigkeit des Antriebssystems weiter erhöht werden.

3.2 Anforderungen des Hauptkunden (MAN)

Die nachfolgende Aufzählung beinhaltet konkrete Punkte, die von Seiten des Hauptkunden MAN für ein zukunftssicheres Antriebssystem gewünscht werden. Die folgenden Punkte wurden wörtlich aus der „Wunschliste“ übernommen:

- (1) Umfassende Diagnose- / Messfunktionen via schneller Ethernet-Schnittstelle unter Einbeziehung des Ethernet-Gesamtkonzeptes der Maschine um z.B. das Referenzsignal der realen Leitachse zu messen, aufzuzeichnen und auszuwerten
- (2) Vorbeugende Diagnose (z.B. Geberausfall in ... Tagen wahrscheinlich)
- (3) „Expertensystem“ soll Fehlerlokalisierung innerhalb max. 10 min sicherstellen und Fehlerbehebung wesentlich vereinfachen
- (4) Webbrowser-Funktionen
- (5) Das Diagnosesystem muss auf mehreren Plattformen laufen können (z.B. MRA-Leitstand)
- (6) Integrierte Datenprotokollierung / -analyse (Registerwerte, Kommandos) muss ohne zusätzliche Hardware (Datenanalyzer) verfügbar sein
- (7) Zyklische Datenprotokollierung auf zentralen Datenbankserver
- (8) Zugriffsmöglichkeit für den Maschinenhersteller auf ausgelieferte Antriebssysteme per Ferndiagnose
- (9) Bedienerführung und Parameterhandling (Konfiguration, Inbetriebnahme, Fehlersuche, Software-Updates) sind wesentlich zu verbessern

3.3 Anforderungen der Fa. Baumüller

Aus der Sicht der Fa. Baumüller Anlagen-Systemtechnik bestehen einer Reihe von Anforderungen, die bei der Entwicklung von BAUDIS NET berücksichtigt werden sollen:

(1) Ausbau der Kompetenz im Bereich der Anlagendiagnose und –überwachung

Die Fa. Baumüller Anlagen-Systemtechnik besitzt mit den Produkten Baudis und Baudis pro langjährige Erfahrung auf dem Gebiet der Ferndiagnose und –überwachung von Antriebssystemen. Bereits das bestehende Produkt Baudis und Baudis pro stellt ein Alleinstellungsmerkmal gegenüber anderen Antriebsherstellern dar. Durch die Entwicklung von BAUDIS NET soll diese Kompetenz weiter ausgebaut werden und so die Wettbewerbsfähigkeit gegenüber den Mitbewerbern erhöht werden. Insbesondere sollen alle bisherigen Funktionen in den Softwarewerkzeugen Baudis und Baudis pro auch in BAUDIS NET enthalten sein.

(2) Berücksichtigung der Wünsche von MAN

Als Hauptkunde kann die MAN die Berücksichtigung ihrer Wünsche im Bereich der Anlagendiagnose und –überwachung erwarten.

(3) Entwicklung von branchenübergreifenden Lösungen

Trotz Berücksichtigung der Wünsche der MAN soll BAUDIS NET für den branchenübergreifenden Einsatz entwickelt werden, so dass ein Einsatz in anderen Branchen (z.B. Werkzeugmaschinen) ohne großen Aufwand möglich ist.

(4) Bereitstellung der Softwarewerkzeuge für die Inbetriebnahme von Antriebssystemen mit zukünftig bis zu 500 Achsen

Antriebssysteme mit mehr als 300 Achsen sind ohne Softwareunterstützung nur noch mit unverhältnismäßig hohem Aufwand in Betrieb zu nehmen. Hierfür soll mit BAUDIS NET ein geeignetes Softwarewerkzeug entwickelt werden.

(5) Verkürzung der Inbetriebnahmezeit und Reduzierung der Inbetriebnahmekosten

Mit Hilfe von BAUDIS NET sollen die Kosten für die Inbetriebnahme durch die Bereitstellung geeigneter softwaregestützter Verfahren langfristig reduziert werden.

(6) Weltweiter Zugriff auf das Antriebssystem zur schnellen und kostengünstigen Diagnose

Für einen schnellen und zuverlässigen Service und zur Anlagendiagnose soll ein weltweiter Zugriff auf das Antriebssystem möglich sein.

(7) Erschließung neuer Dienstleistungen zur Anlagendiagnose und -überwachung

BAUDIS NET kann Grundlage für die Erschließung neuer Service-Dienstleistungen sein. Z.B. können mit geringem Personalaufwand viele große Anlagen hinsichtlich Zustand überwacht werden und im Fehlerfall entsprechendes Servicepersonal informiert werden.

3.4 Einschränkungen von Baudis und Baudis pro

Die beiden Diagnosewerkzeuge Baudis und Baudis pro werden seit einigen Jahren erfolgreich eingesetzt. Die hier gewonnenen Funktionen und Erfahrungen sollen auch in BAUDIS NET zur Verfügung stehen. Leider ist jedoch die Baudis und Baudis pro zu Grunde liegende Softwarestruktur als Basis für die Erfüllung obengenannter Anforderungen nicht geeignet. Im folgenden werden die in Baudis und Baudis pro enthaltenen, strukturimmanenten Einschränkungen aufgelistet, die eine Weiterentwicklung der beiden Softwarewerkzeuge, so dass sie die obengenannten Forderungen erfüllen können, nicht sinnvoll erscheinen lassen:

- Die Bandbreite für die Datenübertragung (RS485 Schnittstellen) reicht nicht aus. Die auf der Bedienoberfläche darzustellenden Informationen können aufgrund der Ring-Kommunikationsstruktur nicht schnell genug übertragen werden. Die Abfrage eines Parameters dauert 50 – 60 ms – bei Anlagen mit ca. 500 Antrieben würde z.B. ein anstehender Fehler erst nach mehr als einer Minute angezeigt werden.
- Jeder Parameter wird einzeln übertragen – Die Übertragung von Parameterpaketen ist nicht möglich
- Zur Visual-Basic-Bedienoberfläche:
 - erfordert Installation
 - Updates sind schwierig durchzuführen
 - Nicht plattformunabhängig
 - Auflösung von 800 x 600 ist fest
 - Das Anlagenbild passt nicht auf eine Seite
- Kein zuverlässiges Backup des Reglerzustands möglich (Firmware und Datensätze)
- Kein Softwareupdate für den G-Drive möglich

- Keine gleichzeitige Behandlung von Antriebsgruppen
- Die Unterstützung bestimmter Sprachen benötigt eine Umschaltung der Betriebssystemsprache
- Die anderen Regler der Baumüller Reglerfamilie werden nicht unterstützt.

3.5 Schlussfolgerungen

Die Ausführungen in den Kapiteln 3.1, 3.2 und 3.3 zeigen, dass an die Struktur von BAUDIS NET umfangreiche und vielfältige Anforderungen gestellt werden müssen. Insbesondere ist es notwendig sehr flexibel und mit verhältnismäßigem Aufwand auf die Anforderungen des Kunden reagieren zu können. Zusätzlich ist heute noch nicht abzusehen, welche Funktionen zukünftig bei der Diagnose vom großen Antriebssystemen vom Kunden gewünscht werden. Dies bedingt, schon beim Entwurf der grundlegende Struktur der Software darauf zu achten, dass eine standardisierte, flexible Struktur entsteht, die leicht durch neue Funktionen erweitert werden kann.

Um diesem hohen Anspruch gerecht zu werden, erscheint eine Aufteilung der Aufgabe in zwei Schritte sinnvoll:

- (1) Schaffung einer zukunftssicheren Infrastruktur für die Anlagendiagnose und Inbetriebnahme.
- (2) Entwicklung von neuen Funktionen für die Anlagendiagnose, -überwachung und Inbetriebnahmeunterstützung.

In den folgenden Kapiteln werden die beiden genannten Punkte näher spezifiziert.

4 Gesamtkonzept BAUDIS NET

4.1 Spezifikation wichtiger Eigenschaften von BAUDIS NET

Im nachfolgenden Kapitel werden wichtige Eigenschaften der BAUDIS NET Infrastruktur sowie wesentliche Funktionen definiert. Dabei liegen die in Kapitel 3 genannten Anforderungen zu Grunde.

4.1.1 Eigenschaften der Infrastruktur

Um eine zukunftssichere und schnelle Infrastruktur für die Diagnose und Inbetriebnahmefunktionen von BAUDIS NET zu erzielen werden einige Eckpunkte spezifiziert:

(1) Verteilte, dezentrale Infrastruktur für die Anlagendiagnose und Inbetriebnahme

Umfangreiche Diagnosefunktionen und Funktionen zur softwaregestützten Inbetriebnahme sollen möglichst nahe am betreffenden Gerät angesiedelt sein. Dies hat den Vorteil, dass viele Aufgaben bei der Diagnose und Inbetriebnahme dezentral ohne große Beanspruchung der Server-Ressourcen zur Verfügung stehen können. Die Verlagerung wesentlicher Funktionalitäten in Richtung der Regler ermöglicht eine wesentlich bessere Verarbeitung großer Datenmengen.

(2) Kommunikations-Infrastruktur basierend auf Ethernet

Industrielles Ethernet ermöglicht im Vergleich zur seriellen Kommunikation über RS485 und das USS-Protokoll eine vielfach höhere Bandbreite für die Datenübertragung. Weiterhin hat sich Ethernet für die Übertragung großer Datenmengen als weitgehender Standard durchgesetzt. Durch die Verwendung von Ethernet mit TCP/IP ist die Kompatibilität von BAUDIS NET mit dem Internet gewährleistet.

(3) Webbasierte Bedienoberflächen

Webbasierte Bedienoberflächen bieten aufgrund ihrer Flexibilität gegenüber konventionellen Oberflächen einige entscheidende Vorteile:

- Die Weboberfläche kann auf beliebigen Client-Rechnern laufen (z.B. Leitstands-PC)
- Eine Installation der BAUDIS NET Bedienoberfläche auf dem Client-Rechner entfällt
- Weboberflächen sind aufgrund ihrer weiten Verbreitung durch das Internet leicht bedienbar
- Die Bedienoberfläche kann mit vertretbarem Aufwand an Kundenwünsche angepasst werden

(4) Plattform-unabhängige Serverarchitektur

In der Anlagenautomatisierung hat sich bis jetzt noch keine der beiden Betriebssystemwelten Linux oder Windows als Standard etabliert. Für BAUDIS NET soll als Betriebssystem für die Serverkomponenten von BAUDIS NET Linux gewählt werden. Dies bietet folgende Vorteile:

- Linux ist ein standardisiertes, offenes Betriebssystem – Windows ist proprietär
- Die erheblichen Lizenzkosten für Serversoftware unter Windows entfallen
- Der Hauptkunde MAN setzt Linux auf den Leitstands-Rechner ein
- Eine Portierung der Serversoftware auf Windows ist bei Verwendung von Linux zusammen mit Java einfach möglich. Bei Entwicklung der Software unter Windows mit einer Sprache aus dem .NET-Framework ist eine Portierung auf Linux nicht möglich.
- Linux ist das meistbenutzte, zuverlässigste Server-Betriebssystem im Internet

(5) Komponentenbasierter Aufbau

Ein komponentenbasierter Aufbau ermöglicht die Erweiterbarkeit mit neuen Funktionen sowie die flexible Anpassung an die Anforderungen anderer Branchen. Durch die konsequente Nutzung von objektorientierter Technologie soll ein hoher Wiederverwendungsgrad der Software erreicht werden.

(6) Einbeziehung der Baumüller Reglerfamilie

Die Infrastruktur von BAUDIS NET soll so entworfen werden, dass alle Regler aus der Reglerfamilie der Fa. Baumüller unterstützt werden können, sofern sie geeignete Schnittstellen für die Übertragung der Diagnosedaten zur Verfügung stellen.

(7) Leichtere Unterstützung vieler Sprachen

Aufgrund der internationalen Geschäftstätigkeit der Fa. Baumüller wird es gefordert die BAUDIS NET Bedienoberfläche in verschiedenen Sprachen auszuliefern. Hierzu ist eine Sprachunterstützung notwendig, die eine Portierung der Bedienoberfläche mit den dazugehörigen Meldungen in eine andere Sprache erlaubt. Die modernen Internettechnologien stellen hierfür geeignete Verfahren zur Verfügung.

(8) Erschließung des Internets für die Anlagendiagnose

Die Grenze zwischen dem firmeninternen Intranet und dem globalen Internet wird in zunehmenden Maße verwischt. Gerade im Bereich der Fernwartung und Ferndiagnose wird zukünftig verstärkt die Forderung bestehen Diagnosedaten über das Internet zu verschicken oder eine Anlagen von einem beliebigen Client im Internet sicher zu überwachen. Durch die Nutzung der Internettechnologien wie z.B. Webseiten, HTML, e-mail etc. sollen diese zukünftigen Möglichkeiten erschlossen werden.

4.1.2 Wesentliche Funktionen in BAUDIS NET

Im Folgenden werden einige wesentlichen neuen Funktionen spezifiziert, die in BAUDIS NET enthalten sein sollen. Die genannten Funktionen sind in zwei Bereiche aufgeteilt: Neue Funktionen im Bereich Inbetriebnahme und Neue Funktionen im Bereich Überwachung und Diagnose. Alle in Baudis und Baudis pro enthaltenen Funktionen sollen weiterhin in BAUDIS NET verfügbar sein. Alle genannten Funktionen werden in den entsprechenden Kapiteln weiter spezifiziert.

Neue Funktionen im Bereich Inbetriebnahme

(1) Softwaregestütztes Firmwareupdate

Im Gegensatz zum manuellen Einspielen der Firmware auf den Regler (z.B. M-Drive) soll die Funktion „softwaregestütztes Firmwareupdate“ ein schnelles und einfaches Bespielen der Regler mit der Firmware ermöglichen. Hierdurch kann besonders bei großen Anlagen eine erhebliche Zeit- und Kosteneinsparung erreicht werden.

(2) Sicheres Backup eines Antriebs

Bei Arbeiten an der Konfiguration eines Reglers ist eine Funktion zum zuverlässigen Backup eines Antriebs erforderlich. Vor der Durchführung der Konfigurationsarbeiten oder einer neuen Reglerfirmware kann eine Sicherung des kompletten vorherigen Antriebszustands erstellt werden.

(3) Funktionen für den automatisierten Antriebtest

Zur Standardisierung und Vereinfachung der Antriebtests sollen softwaregestützte Funktionen zur Verfügung gestellt werden. Z.B. sollen automatisierte Funktionsprüfungen den Antrieb in verschiedenen Betriebszuständen überprüfen und ein automatisiertes Prüf- und Abnahmeprotokoll erstellt werden.

(4) Softwaregestützte Antriebsoptimierung

BAUDIS NET soll Funktionen zur softwaregestützten Optimierung der Reglereinstellungen zur Verfügung stellen.

Neue Funktionen im Bereich Diagnose und Überwachung

(5) Erweiterte Funktionen für die Aufzeichnung von Parametern

Im Gegensatz zu den bisherigen Möglichkeiten soll BAUDIS NET die Möglichkeit bieten beliebige Gruppen von Parametern von beliebigen Reglern in einer Aufzeichnung über einen festgelegten Zeitraum aufzuzeichnen. Zusätzlich können mehrere Aufzeichnungen gleichzeitig gestartet werden.

(6) Erweiterte Funktionen für die Visualisierung von Aufzeichnungen oder Parametern
BAUDIS NET soll umfangreiche Möglichkeiten bieten Aufzeichnungen oder Parameterverläufe in verschiedenen Darstellungen zu visualisieren.

(7) Vorbeugende Diagnose

Mit Hilfe von konfigurierbaren Routineüberwachungen wichtiger Betriebsparameter des Antriebssystems soll die Zuverlässigkeit der Anlagen weiter erhöht werden. Wichtige Überwachungsinformationen sollen automatisch dem zuständigen Personal weitergeleitet werden.

(8) Wesentlich erweiterte Diagnosemöglichkeiten zur Fehlersuche

Durch die dezentrale Verteilung von Diagnosefunktionen sollen in BAUDIS NET wesentlich erweiterte Diagnosemöglichkeiten zur Fehlersuche zur Verfügung gestellt werden. Z.B. sollen sehr variable Triggerbedingungen für die Aufzeichnung sowie Skriptunterstützung auf dem dem Regler beigelegten Kommunikations-PC zur Verfügung stehen.

4.2 Überblick Grobstruktur BAUDIS NET

Ziel der Entwicklung

BAUDIS NET soll die wesentliche Funktionalität, die zur Konfiguration, Bedienung, Diagnose und Inbetriebnahmen von einer großen Anzahl von Antrieben notwendig ist, umfassen. BAUDIS NET soll ein komponentebasiertes und erweiterbares System sein, um es an zukünftige Anforderungen anzupassen.

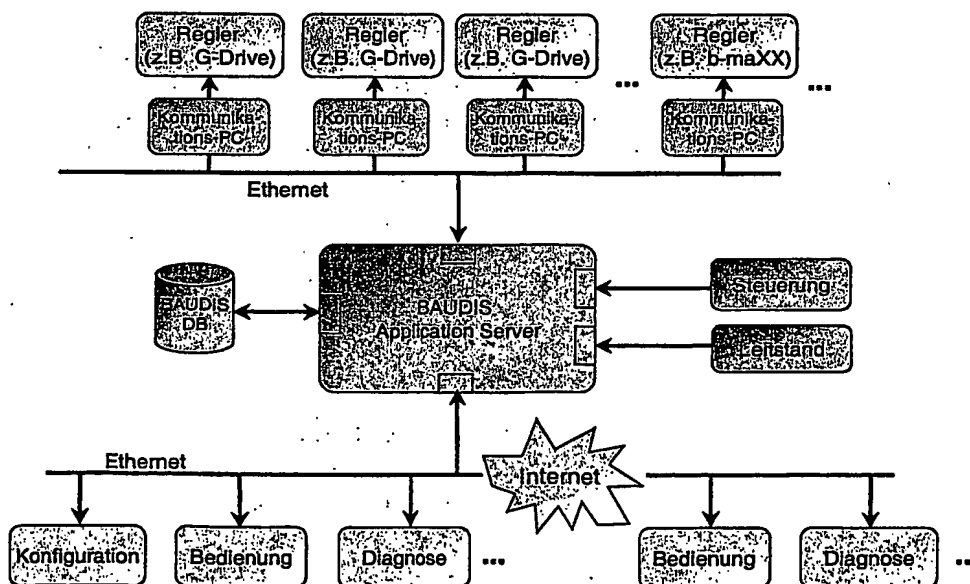


Abbildung 4-1: Grobstruktur von Baudis NET

Die obige Abbildung 4-1 gibt einen Überblick über die Grobstruktur von Baudis NET. Dem Anwender stehen verschiedene web-basierte Bedienoberflächen zur Verfügung, die ihm die Funktionalität von BAUDIS NET in einer für ihn geeigneten Darstellung präsentieren. Für die Bedienung des Systems ist es unerheblich, ob sich der Benutzer lokal an der Anlage oder an einem anderen Ort befindet.

Die vom Benutzer gewünschten Funktionen werden von den Benutzeroberflächen an den Application-Server weitergeleitet. Hier ist jede Funktionalität, die in der Oberfläche zur Verfügung steht, implementiert. Weiter übernimmt der Baudis Application-Server die Speicherung von allen anfallenden Daten in der Baudis-Datenbank. Alle anlagenspezifischen Daten wie z.B. die Anlagenkonfiguration oder Bauteildatenbanken, alle diagnosespezifischen Daten wie z.B. Langzeitaufzeichnungen oder Ringspeicherdaten sowie anwendungsbezogene Daten werden in der Baudis Datenbank verwaltet.

Werden von der Steuerung oder vom Leitstand einer Druckmaschine für die Diagnose relevante Informationen zur Verfügung gestellt, können sie durch spezielle in den Application Server integrierte Komponenten weiterverarbeitet werden.

Die von der Bedienoberfläche am Application-Server eingehenden Aufträge werden dort verarbeitet und in für die entsprechenden Regler verständliche Befehle umgesetzt. Die Kommunikation zwischen Application Server und Kommunikations-PC mit angeschlossenem Regler erfolgt über Ethernet und darauf aufsetzende geeignete Protokolle. Am Kommunikations-PC werden die vom Application Server empfangenen Aufträge ausgeführt und das Ergebnis an den Application Server zurückgeschickt.

Die Abbildung 4-1 zeigt verschiedene Reglertypen aus der Baumüller Reglerfamilie: G-Drive und b-maXX. Jedem Regler wird ein Kommunikations-PC zugeordnet. Um zukünftig auch einen b-maXX Regler in die Diagnose einbeziehen zu können, kann ein als BACI-Einsteckkarte realisierter Kommunikations-PC in den b-maXX gesteckt werden und darüber die notwendigen Diagnosedaten an den BAUDIS NET Server geschickt werden. Dies ist jedoch nicht Bestandteil des vorliegenden Entwicklungsprojekts BAUDIS NET und kann in einem separaten Projekt realisiert werden.

Für Reglertypen, die keine BACI-Einsteckkarten aufnehmen können, kann die Verarbeitung der Aufträge auch durch andere Mechanismen sichergestellt werden.

Die in der Anforderungsanalyse aus Kapitel 3 genannten Anforderungen bedingen eine komponentenbasierte, verteilte Architektur von BAUDIS NET. Gemäß den allgemeinen Grundsätzen der Softwareentwicklung werden die Datenerfassung, die Datenverarbeitung und -speicherung und die Benutzeroberflächen modularisiert und voneinander getrennt. Dadurch wird eine übersichtliche und besser skalierbare Struktur erreicht, die leicht um weitere Funktionalitäten erweitert werden kann. Hierdurch kann gewährleistet werden, dass BAUDIS NET mit steigender Anzahl von Antrieben „mitwächst“ (Skalierbarkeit). Durch die Verwendung von Ethernet und TCP/IP für die Kommunikation zwischen den Kommunikations-PCs, dem Baudis Application Server und den Anwendungen steht eine wesentlich höhere Bandbreite zur Datenübertragung zur Verfügung. Dies resultiert in einem im Vergleich zu Baudis und Baudis pro wesentlich schnelleren Diagnosesystem.

Weiter erleichtert die komponentenbasierte Struktur den großen Funktionsumfang von BAUDIS NET abzudecken. Für jede Benutzergruppe kann eine eigens auf die Bedürfnisse zugeschnittene Bedienoberfläche entwickelt werden, die auf die darunter liegende Infrastruktur (Application Server) zugreift.

Durch die Trennung von Bedienoberfläche und Implementierung der Funktionalität im Application Server können zukünftig mit weniger Aufwand neue Anwendungen entwickelt werden. Durch Nutzung moderner Softwaretechnologien kann das Internet als weltweit verfügbares und akzeptiertes Kommunikationsmedium genutzt werden. Damit ist es unerheblich, ob die Überwachung der Anlage lokal vor Ort oder von einem anderen Ort, wie z.B. dem Servicebüro durchgeführt wird. Durch die Nutzung von gängigen Internetbrowsern für die Benutzeroberflächen wird der Installationsaufwand beim Anwender wesentlich reduziert und die Hürde für den Anwender das Diagnosesystem zu benutzen deutlich herabgesetzt.

Die komponentenbasierte Architektur ermöglicht weiterhin die Unterstützung von neu entwickelten Verfahren zur Überwachung und Diagnose von Antrieben, indem neue Funktionen als Komponente in den Application-Server eingebunden werden.

4.3 Kundennutzen

Das folgende Kapitel listet wesentliche Vorteile von BAUDIS NET aus der Sicht des Kunden auf:

(1) Der Kunde hat durch die Weboberfläche einen universellen Zugriff auf die Diagnosefunktionen:

- Die richtige Information zeitnah am richtigen Ort
- Einfache Bedienoberfläche durch Webbrowser
- Benutzerführung erleichtert die Bedienung und Konfiguration
- Die Web-Oberfläche ist plattformunabhängig
- Bedienung über das Internet möglich, sofern gewünscht

(2) Der Kunde bekommt für ihn aufbereitete Informationen über den Zustand der Anlage:

- Informationsaufbereitung in Form von graphische Darstellungen
- Vorbeugende Diagnose

(3) Die wesentlich erweiterten Diagnosemöglichkeiten ermöglichen:

- Erweiterte Überwachung der Anlage
- Einfachere Lokalisierung der Ursache im Fehlerfall

(4) Die Funktionen zur Inbetriebnahmeunterstützung ermöglichen:

- Schnellere Inbetriebnahme → Kostenreduzierung
- Erhöhte Qualität der Inbetriebnahme durch definierte und dokumentierte Abnahmeprotokolle

5 Spezifikation der Bedienoberflächen

Verschiedene Bedienoberflächen

Wie bereits in Kapitel 1 erwähnt, bietet BAUDIS NET die Möglichkeit verschiedene, auf die speziellen Bedürfnisse der Benutzergruppe angepasste, Bedienoberflächen zur Verfügung zu stellen. In der ersten Ausbaustufe sollen zwei verschiedene Bedienoberflächen mit verschiedenen Zielrichtungen entwickelt werden. Die aus Baudis und Baudis pro bekannten Funktionen sollen auch in BAUDIS NET zur Verfügung stehen.

Kapitel 5.1 spezifiziert die Bedienoberfläche für die Anlageninbetriebnahme, -überwachung und Diagnose. Kapitel 5.2 spezifiziert eine Bedienoberfläche zur Konfiguration der Anlage in BAUDIS NET. Eine weitergehende Spezifikation steht noch aus.

5.1 Bedienoberfläche für die Anlageninbetriebnahme, – überwachung und Diagnose

Für die Inbetriebnahme, Überwachung und Diagnose von großen Antriebssystemen soll eine Bedienoberfläche zur Verfügung gestellt werden, die die dargestellten Informationen graphisch aufbereitet dem Anwender zur Verfügung stellt. Die Diagnose-Informationen sollen einem breitem Benutzerkreis zur Verfügung gestellt werden. Z.B. soll die zu entwickelnde Anwendung unter anderem auch am Leitstand einer Druckmaschine ausgeführt werden. Weiter kann sie eine Informationsquelle sein, mit der sich das technische Personal oder das Management einen schnellen Überblick über die Funktion der Maschine beschaffen kann, sowie vom Baumüller Personal während der Inbetriebnahmephase grundlegende Konfigurationsarbeiten durchgeführt werden können. Die Benutzung der Funktionen erfordern eine Benutzerauthentifizierung.

Um die Informationen auf der Benutzeroberfläche ständig aktuell zu halten, muss ein Mechanismus implementiert werden, mit dem die aktuellen Daten ständig vom Server abgefragt werden. Nur so kann sichergestellt werden, dass der Benutzer über auftretenden Fehler oder andere Ereignisse zeitnah informiert wird.

5.1.1 Zielgruppe

Personal des Kunden, z.B. Drucker, Techniker etc. und das Service – und Inbetriebnahmepersonal der Fa. Baumüller Anlagen-Systemtechnik.

5.1.2 Mehrsprachigkeit

Nachdem die Anwendung zur Diagnose und Anlagenüberwachung hauptsächlich vom Personal des Kunden benutzt werden soll, ist die Unterstützung von verschiedenen Sprachen notwendig.

5.1.3 Implementierung

Aufgrund des breiten Benutzerkreises muss die Anwendung sehr leicht für viele Anwender zugänglich sein. Hier bietet sich die Implementierung als Weboberfläche die in einem Webbrowser wie z.B. Internet Explorer oder Netscape Navigator läuft an. Der Zugriff soll sowohl über das Internet als auch über Einwahlverbindung oder aus dem lokalen Netz möglich sein. Die automatische Aktualisierung der Oberfläche muss mit Hilfe eines geeigneten Mechanismus realisiert werden.

5.1.4 Benutzer-Authentifizierung

Die nachfolgend spezifizierten Funktionsgruppen sind zu drei Benutzergruppen zugeordnet: Benutzergruppe Drucker, Benutzergruppe Techniker und die Benutzergruppe Inbetriebnehmer. Die Benutzergruppe Drucker besitzt nur Zugriff auf die für sie wesentlichen Funktionen. Der Benutzergruppe Techniker stehen sowohl alle Funktionen der Benutzergruppe Drucker sowie weitere Funktionen zur Verfügung. Die Benutzergruppe Inbetriebnehmer besitzt vollen Zugriff auf alle Funktionen.

Die Einteilung der Benutzer in die obengenannten Gruppen ist jedoch nur als Richtlinie zu verstehen, denn es kommt in der Praxis oft vor, dass einem Benutzer temporär oder dauerhaft Zugriff auf spezielle Funktionen gegeben werden soll. Dies kann durch die Erstellung einer neuen Benutzergruppe erreicht werden, in der die gewünschten Funktionen freigeschaltet bzw. gesperrt sind. Die Funktionen zur Erstellung einer neuen Benutzergruppe ist vorerst nicht Teil dieser Spezifikation und soll nur von eingewiesenem Personal durchgeführt werden.

5.1.5 Spezifizierte Funktionalitäten für die Benutzergruppe „Drucker“

Die nachfolgend spezifizierten Funktionen sind in Funktionsgruppen und Funktionen gruppiert. Die Abbildung 5-1 zeigt ein UML-Anwendungsfalldiagramm aus dem die wesentlichen Bedienfunktionen hervorgehen.

5.1.5.1 Funktionsgruppe Sprachenauswahl

Mit der Funktionsgruppe Sprachenauswahl kann die gewünschte Sprache jederzeit gewählt werden. Alle Webseiten sollen in der gewünschten Sprache angezeigt werden. Die Daten der Anlage (wie z.B. das Anlagenbild) sollen in verschiedenen Sprachen auf dem Application Server zur Verfügung stehen. Folgende Aktionen müssen möglich sein:

5.1.5.1.1 Sprache auswählen

5.1.5.2 Funktionsgruppe Benutzerauthentifizierung

Die Funktionen der Funktionsgruppe Benutzerauthentifizierung ermöglichen einen Wechsel der Benutzeridentität. Für den Zugriff auf die Funktionen der Benutzergruppe Drucker ist keine personalisierte Anmeldung erforderlich. Bei Bedarf kann sie jedoch eingeschaltet werden. Die Daten der An- und Abmeldung werden in der Datenbank gespeichert, um später kritische Aktionen nachvollziehen zu können. Folgende Aktionen sollen möglich sein:

5.1.5.2.1 Benutzer anmelden

Der Benutzer kann seinen Benutzernamen und sein Passwort angeben, um die seiner Benutzergruppe entsprechenden Rechte zusammen mit der entsprechenden Benutzeroberfläche zu erhalten.

5.1.5.2.2 Benutzer abmelden

Der Benutzer meldet sich ab.

5.1.5.3 Funktionsgruppe Logbuch

Mit Hilfe der Funktionsgruppe Logbuch sollen alle an den Antrieben auftretenden Ereignisse aufgezeichnet werden. Mit komfortablen Such- und Filterfunktionen können die Informationen des Logbuchs nach vorgegebenen Kriterien durchsucht, sortiert und angezeigt werden. Zusätzlich kann eine Beschreibung des Ereignisses angezeigt werden. Folgende Aktionen müssen möglich sein:

5.1.5.3.1 Logbuch ansehen

Das Logbuch speichert die aufgetretenen Ereignisse. Folgende Informationen sollen angezeigt werden:

- Ereignistyp
- Ereignis kommt
- Ereignis geht
- Antrieb
- Ereignis-Nummer
- Meldung
- Betriebsart
- Maschinengeschwindigkeit
- Sicherhalt?
- Reglerfreigabe?

5.1.5.3.2 Logbuch filtern und sortieren

Das Logbuch soll nach folgenden Kriterien gefiltert oder sortiert werden:

- Filtern nach Antrieben

- Filtern nach Ereignistypen
- Filtern nach Datum und Uhrzeit

5.1.5.3.3 Logbuch drucken

Das Logbuch soll entweder gefiltert oder sortiert oder im ganzen in einem tabellarischen Textformat auf dem Standarddrucker ausgegeben werden können.

5.1.5.4 Funktionsgruppe Ereignisanzeige

In der Funktionsgruppe Ereignisanzeige werden dem Benutzer die aktuellen Ereignisse (z.B. Fehler) tabellarisch angezeigt. Von hier aus kann er eine Beschreibung des aufgetretenen Ereignisses abrufen. Einige Ereignisse, z.B. solche, die nicht am Leitstand quittiert werden, müssen vom Benutzer quittiert werden, so dass sie aus der Ereignisanzeige verschwinden.

5.1.5.4.1 Ereignis quittieren

Das Ereignis wird quittiert und verschwindet aus der Ereignisanzeige.

5.1.5.5 Funktionsgruppe Anlagenbild

In der Funktionsgruppe Anlagenbild bekommt der Anwender einen graphischen Überblick über die Anlage präsentiert. Hierbei sind verschiedene graphische Übersichten vorgesehen, die bei Bedarf erweitert werden können. Grundsätzlich sollen die anzuzeigenden Informationen in drei Ebenen eingeteilt werden:

1. Ebene: Anlagenebene

In der Anlagenebene erhält der Benutzer einen Überblick über die gesamte Anlage. Hier stehen ihm in der ersten Ausbaustufe zwei verschiedene Ansichten zur Verfügung: In einer Gesamtdarstellung kann die gesamte Anlage in Form einer an die Realität angelehnte Symboldarstellung betrachtet werden. Diese Ansicht wird „Anlagenübersicht“ genannt. In einer zweiten Ansicht bekommt der Benutzer einen Überblick über die Struktur des Antriebssystems. Diese Ansicht wird „Antriebssystem“ genannt.

Grundsätzlich kann der Benutzer von der Anlagenebene aus durch Klicken auf eines der Symbolelemente in die darunterliegende Antriebsgruppen-Ebene springen.

2. Ebene: Antriebsgruppen- Ebene

In der zweiten Ebenen bekommt der Benutzer verschiedene Ansichten zur Auswahl angeboten, die sich auf die jeweils gewählte Antriebsgruppe beziehen. In der ersten Ausbaustufe werden hier Diagnose-Informationen zu einer Antriebsgruppe, wie z.B. Ströme, Temperatur und Geschwindigkeit angezeigt. Zusätzlich können von hier aus weitere Diagnosefunktionen aktiviert werden. (siehe z.B. Kapitel 5.1.5.5.2) Weitere Ansichten sind konfigurierbar.

Von dieser Ebene aus können die Ansichten der 3. Ebene aufgerufen werden.

3. Ebene: Antriebsebene

In der dritten Ebene bekommt der Benutzer verschiedene Ansichten zu einem einzelnen Antrieb präsentiert. Je nach Typ des Gerätes werden entsprechende Ansichten zur Verfügung gestellt. Beispiele für die verschiedenen Ansichten können dem Oberflächenprototyp entnommen werden.

5.1.5.5.1 Anlagenbild navigieren

Mit dieser Funktion kann in den oben genannten Ebenene navigiert werden.

5.1.5.5.2 Überwachung für eine Antriebsauswahl starten

Mit dieser Funktion kann für eine Antriebsgruppe eine kontinuierliche Überwachung festgelegter Parameter gestartet werden. Weiteres hierzu findet sich in Kapitel 5.1.5.6.

5.1.5.6 Funktionsgruppe Überwachung

Mit Hilfe der Funktionsgruppe Überwachung soll der Drucker wichtige Parameter von einem oder einer Gruppe von Antrieben visualisieren können. Dabei stehen ihm eine Auswahl von vordefinierten Überwachungsansichten zur Verfügung, die er mit Hilfe des Kontextmenüs und der Aktion 5.1.5.5.2 aufrufen kann. Folgenden Überwachungsansichten sollen mindestens zur Verfügung stehen:

- Motortemperatur
- Ströme
- Geschwindigkeit

Folgende Aktionen müssen möglich sein:

5.1.5.6.1 Auswahl einer Überwachungsansicht

Der Anwender wählt eine vordefinierte Überwachungsansicht aus. Ein Fenster wird geöffnet, in dem die ausgewählten Informationen in Form eines xy-Plots dargestellt werden. Die Grafik wird in vordefinierten Abständen ständig aktualisiert. Je nach der Anzahl der ausgewählten Antriebe werden ein oder mehrere Graphen im xy-Plot dargestellt.

5.1.5.6.2 Überwachungsansicht speichern

Der Benutzer kann die gesamten seit dem Start der Überwachungsansicht angefallenen Daten als Datei abspeichern. Die Daten werden im Standardformat für Aufzeichnungen abgespeichert.

5.1.5.6.3 Überwachungsansicht drucken

Druckt die aktuelle Überwachungsansicht.

5.1.5.6.4 Überwachungsansicht offline schalten

Unterbricht die Aktualisierung der Überwachungsansicht. Dies wird z.B. beim Drucken benötigt.

5.1.5.6.5 Überwachungsansicht beenden

Beendet die Aktualisierung der Überwachungsansicht und schließt das Fenster.

5.1.5.7 Funktionsgruppe Aufzeichnung

In der Funktionsgruppe Aufzeichnung kann der Drucker aus einer Liste mit vorkonfigurierten Aufzeichnungen wählen und diese Starten und Stoppen. Zusätzlich bekommt er angezeigt, welche Aufzeichnungen gerade aktiv sind. Folgende Aktionen sollen möglich sein:

5.1.5.7.1 Aufzeichnung auswählen

Der Anwender wählt aus einer Liste eine Aufzeichnung aus, die er starten oder beenden will.

5.1.5.7.2 Aufzeichnung starten

Der Anwender startet die ausgewählte Aufzeichnung. Dabei können nur solche Aufzeichnungen gestartet werden, die nicht schon laufen.

5.1.5.7.3 Aufzeichnung stoppen und speichern

Der Anwender beendet eine laufende Aufzeichnung und speichert sie in der Datenbank ab.

5.1.5.8 Funktionsgruppe Dokumentation

In der Funktionsgruppe Dokumentation hat der Anwender Zugriff auf alle verfügbaren Dokumente. Die Dokumentation zu den einzelnen Anlagenteilen, die Fehlermeldungen und Servicebeschreibungen sind in der Baudis Site abgelegt. Hier kann er in den Dokumenten navigieren und suchen. Zu den Fehlerbeschreibungen können beliebige Kommentare hinzugefügt oder gelöscht werden. Folgende Aktionen sollen möglich sein:

5.1.5.8.1 In Dokumentation navigieren und suchen

Der Anwender navigiert in den verschiedenen Dokumenten.

5.1.5.8.2 Ereignisbeschreibung anzeigen

Der Anwender lässt sich zu einem Ereignis mit einer Ereignisnummer eine Beschreibung anzeigen.

5.1.5.8.3 Kommentar zu einer Ereignisbeschreibung hinzufügen

Der Anwender fügt einen Kommentar zu einer Ereignisbeschreibung hinzu oder ergänzt einen bestehenden.

5.1.5.8.4 Kommentar zu einer Ereignisbeschreibung löschen

Der Anwender löscht einen Kommentar. Dies kann jeder Anwender mit jedem Kommentar tun, ein Passwortschutz ist nicht vorgesehen.

5.1.6 Spezifizierte Funktionalitäten für die Benutzergruppe „Techniker“

Die nachfolgend spezifizierten Funktionen sind in Funktionsgruppen und Funktionen gruppiert und spezifizieren die Aktionen, die die Benutzer der Benutzergruppe Techniker ausführen können. Jeder Benutzer der Benutzergruppe Techniker erbt alle Funktionen der Benutzergruppe Drucker.

Die Abbildung 5-3 zeigt ein Anwendungsfalldiagramm aus dem die vorgesehenen Bedienfunktionen hervorgehen. Zusätzlich zu den in dieser Abbildung genannten Funktionen stehen dem Techniker alle Funktionen aus Abbildung 5-2 zur Verfügung.



5.1.6.1 Funktionsgruppe Logbuch

Die Funktionsgruppe Logbuch ist bereits in Kapitel 5.1.5.3 beschrieben. Zusätzlich zu den dort genannten Funktionen kann der Techniker das Logbuch in eine Datei auf dem Client-Rechner exportieren und nicht mehr benötigte Logbucheinträge archivieren. Folgende Aktionen sollen möglich sein:

5.1.6.1.1 Logbuch exportieren

Das aktuell angezeigte Logbuch wird in eine Datei auf dem Clientrechner geschrieben. Dabei kann es sich um das komplette Logbuch oder um einen mit Hilfe der Filterfunktion reduzierten Teil des Logbuchs handeln.

5.1.6.1.2 Logbuch archivieren

Um die Länge des Logbuchs zu begrenzen, kann der Techniker nicht mehr benötigte Einträge filtern und archivieren. Nach dem Archivieren von Einträgen werden diese im Logbuch nicht mehr angezeigt.

5.1.6.2 Funktionsgruppe Überwachung

Die Funktionsgruppe Überwachung ist bereits in Kapitel 5.1.5.6 beschrieben. Zusätzlich zu den dort genannten Funktionen kann der Techniker für alle Benutzer von BAUDIS NET neue Überwachungsansichten erstellen, vorhandene editieren und löschen. Die standardmäßig vorhandenen Überwachungsansichten aus Kapitel 5.1.5.6 können weder editiert noch gelöscht werden. Will der Anwender eine dieser Überwachungsansichten verändern muss er diese kopieren und danach editieren. Folgende Aktionen sollen möglich sein:

5.1.6.2.1 Überwachungsansicht erstellen

Der Anwender erstellt eine neue Überwachungsansicht. Hierbei wird ein Konfigurationswizard aufgerufen, der die notwendigen Informationen vom Benutzer abfragt.

5.1.6.2.2 Überwachungsansicht kopieren

Um eine bestehende Überwachungsansicht zu verändern kann der Anwender ein Kopie erstellen und diese danach bearbeiten.

5.1.6.2.3 Überwachungsansicht löschen

Eine von einem Benutzer erstellte Überwachungsansicht wird gelöscht.

5.1.6.2.4 Überwachungsansicht editieren

Beim Editieren einer Überwachungsansicht kann der Anwender durch den Konfigurations-Wizard blättern und Einstellungen verändern.

5.1.6.3 Funktionsgruppe Parameteranzeige

Die Funktionsgruppe Parameteranzeige visualisiert beliebige Parameter von einem oder mehreren Reglern. Aus einer Parameterliste kann der Anwender in Abhängigkeit vom Reglertyp die zur Verfügung stehenden Parameter auswählen und online in einer Tabelle darstellen. In Anlehnung an Baudis und Baudis pro können Parametergruppen zusammengestellt werden. Jede Parametergruppe besteht aus einer oder mehreren Parameterseiten. Jede Parameterliste kann mehrere Parameter von beliebigen Reglern enthalten. Der Anwender kann neue Parametergruppen erstellen und vorhandene kopieren und löschen. Jede Parametergruppe kann jederzeit verändert werden. Die Änderungen werden jedoch erst nach dem Speichern der Parametergruppe wirksam. Folgende Aktionen sollen möglich sein:

5.1.6.3.1 Parametergruppe auswählen

Der Anwender wählt eine Parametergruppe aus einer Liste aus. Auf der Oberfläche erscheinen die in der Parametergruppe enthaltenen Parameterseiten in Form von Karteireitern zur Auswahl.

5.1.6.3.2 Parameterseite auswählen

Durch Klicken auf die Karteireiter wählt der Anwender die gewünschte Parameterseite aus.

5.1.6.3.3 Parameter-Aktualisierung starten

Durch Start der Parameter-Aktualisierung werden alle Parameter auf der angezeigten Parameterseite zyklisch aktualisiert.

5.1.6.3.4 Parameter-Aktualisierung stoppen

Beendet die zyklische Parameter-Aktualisierung.

5.1.6.3.5 Antrieb auswählen

Um einen Parameter anzuzeigen muss zunächst eine Antrieb ausgewählt werden. Durch die Auswahl des Antriebs ist gleichzeitig der Reglertyp bekannt. In „Parameter auswählen“ aus Kapitel 5.1.6.3.6 wird nur die Parameterliste des entsprechenden Reglertyps angezeigt.

5.1.6.3.6 Parameter auswählen

Um einen Parameter auf einer Parameterseite anzuzeigen, muss der Anwender die Parameternummer des gewünschten Parameters wählen. Dies geschieht mit einem Parameterfenster, das verschiedenen Auswahlmechanismen bietet.

5.1.6.3.7 Index auswählen

Der Anwender wählt den Index des Parameters aus.

5.1.6.3.8 Parameter schreiben

Der Anwender beschreibt einen Parameter.

5.1.6.3.9 Parameter-Attribut anzeigen

Das Attribut des Parameters wird in einem kleinen Fenster aufgeschlüsselt angezeigt.

5.1.6.3.10 Parametergruppe erstellen

Der Anwender kann eine Parametergruppe neu erstellen. Mit Hilfe eines Konfigurations-Wizard werden die notwendigen Informationen vom Benutzer abgefragt.

5.1.6.3.11 Parametergruppe kopieren

Um eine Parametergruppe zu verändern kann sie zunächst kopiert werden und dann editiert werden.

5.1.6.3.12 Parametergruppe löschen

Löscht eine Parametergruppe

5.1.6.3.13 Parametergruppe speichern

Speichert eine veränderte Parametergruppe.

5.1.6.3.14 Einen Parameter visualisieren

Mit Hilfe dieser Funktion wird der gewünschte Parameter dem Visualisierungsmodul übergeben und graphisch dargestellt. Für die Konfiguration des Grafik kann eine Stan-

dardkonfiguration gewählt werden, oder mit Hilfe eines Konfigurations-Wizards eine benutzerdefinierte Grafik erstellt werden.

5.1.6.4 Funktionsgruppe Aufzeichnung

In der Funktionsgruppe Aufzeichnung kann der Werteverlauf von einer Anzahl von beliebigen Parametern auf beliebigen Reglern aufgezeichnet werden.

Eine Aufzeichnung fordert von einem oder mehreren Antrieben einen oder mehrere Parameter an und legt sie zusammen mit einem Zeitstempel in einer Datenbank ab. Konfigurierbar sind zusätzlich das Zeitintervall mit dem die Parameter von den Antrieben gelesen werden, die Dauer der Aufzeichnung, ein Name und eine Beschreibung der Aufzeichnung.

Eine Aufzeichnung kann manuell oder beim Eintreten einer Triggerbedingung gestartet und/oder beendet werden.

Es gibt verschiedene Typen von Aufzeichnungen: Aufzeichnungen, die im Ringspeicher des G-Drives abgelegt werden und somit nur verfügbar sind sofern der Regler ein G-Drive ist. (Ringspeicheraufzeichnung) Weiter gibt es Aufzeichnungen, die auf dem BAUDIS NET Server abgelegt werden. (Baudisaufzeichnung) Bei Baudisaufzeichnungen unterscheidet man

zwei Typen: Typ 1 speichert alle Daten dauerhaft, Typ 2 speichert jeweils nur die Daten eines vordefinierten Zeitabschnitts. (z.B. die Daten eines Tages)

Beide Typen von Aufzeichnungen sollen aus der Sicht des Anwenders äquivalent zu bedienen sein. Lediglich bei der Konfiguration der Aufzeichnung muss entschieden werden, ob es sich um eine Ringspeicheraufzeichnung oder um eine Baudisaufzeichnung handelt.

Die Konfiguration einer Aufzeichnung soll über Konfigurations-Wizards erfolgen.

Folgende Aktionen müssen möglich sein:

5.1.6.4.1 Aufzeichnungskonfiguration erstellen

Mit Hilfe eines Konfigurations-Wizard soll der Anwender eine Konfiguration einer Aufzeichnung (Aufzeichnungskonfiguration) erstellen können. Nach Abschluss der Konfiguration wird die Aufzeichnungskonfiguration gespeichert und steht zum Start zur Verfügung.

5.1.6.4.2 Aufzeichnungskonfiguration editieren

Der Benutzer wählt eine existierende Aufzeichnungskonfiguration und verändert dessen Eigenschaften.

5.1.6.4.3 Aufzeichnungskonfiguration kopieren

Der Benutzer wählt eine Aufzeichnungskonfiguration und erstellt eine Kopie.

5.1.6.4.4 Aufzeichnungskonfiguration löschen

Der Benutzer löscht eine Aufzeichnungskonfiguration.

5.1.6.4.5 Exportieren einer Aufzeichnung

Nach erfolgter Aufzeichnung kann eine Aufzeichnung in Form einer Datei auf dem Clientrechner exportiert werden.

5.1.6.4.6 Aufzeichnung stoppen und verwerfen

Bei der Fehlersuche kann z.B. eine Aufzeichnung gestartet werden, um ein eventuell auftretendes Ereignis aufzuzeichnen. Falls dieses Ereignis während der Aufzeichnungszeit nicht eingetreten ist, kann die Aufzeichnung gestoppt und gleichzeitig verworfen werden. Dadurch kann das auf dem Server abgespeicherte Datenvolumen re-

duziert werden, denn einmal gestoppte Aufzeichnungen werden meist nachträglich nicht mehr gelöscht.

5.1.6.5 Funktionsgruppe Antriebe sichern (nur G-Drive)

Die Funktionsgruppe Antriebe sichern ermöglicht die komplette Sicherung eines G-Drives. Hierbei wird die Firmware zusammen mit dem Parameterdatensatz von der Flashkarte in einer Datenbank auf dem BAUDIS NET Server gesichert. Alle Aktionen werden in einem Logfile protokolliert. Folgende Aktionen müssen möglich sein:

5.1.6.5.1 Sicherung erstellen

Der Benutzer kann aus einer Liste von Antrieben auswählen, welche Antriebe in die Sicherung mit einbezogen werden sollen. In der Liste sind nur Antriebe gezeigt, für die die Sicherungsfunktion verfügbar ist. (z.Zt. nur G-Drive) Zu der Antriebsauswahl kann noch ein Kommentar hinzugefügt werden. Zusätzlich werden der Benutzer, der die Sicherung erstellt, und ein Zeitstempel erfasst.

5.1.6.5.2 Sicherung starten

Das Sichern der Firmware und der Datensätze für die ausgewählten Antriebe wird gestartet. Alle Aktionen werden in einem Log-File protokolliert.

5.1.6.5.3 Sicherung zurückladen

Eine bestehende Sicherung wird auf die Antriebe zurückgespeichert.

5.1.6.5.4 Sicherung auswählen

Vor der Aktion „Sicherung zurückspeichern“ muss der Anwender eine Sicherung aus einer Liste mit auf dem BAUDIS NET Server abgelegten Sicherungen auswählen.

5.1.6.6 Funktionsgruppe Visualisierung

Die Funktionsgruppe Visualisierung ermöglicht die graphische Darstellung beliebiger Aufzeichnungen oder Parameter. Der Anwender soll sich verschiedene Darstellungen mit Hilfe eines Visualisierungs-Wizards frei konfigurieren können. Folgende Fähigkeiten soll das Visualisierungsmodul besitzen:

- Verschiedene Diagrammtypen (xy-chart, 3D-Chart, Balkendiagramm, Tortendiagramm, SPC-Chart, Bitleistendarstellung)
- Verschiedene Interpolationsarten
- Mehrere Graphen in einem Diagramm anzeigen
- Mehrere Skalen mit verschiedenen Einheiten
- Automatische oder manuelle Auswahl des angezeigten Bereiches (Skalierung)
- Zoom
- Beliebige Beschriftungen der Graphen und Achsen
- Anzeige eines Gitters (Grid)
- Anzeige von Hilfslinien
- Freie Farbwahl für die einzelnen Diagrammelemente
- Gewichtung der Daten mit Wichtungsfaktoren
- Datencursor zur Anzeige des exakten Wertes an einer Stelle des Graphen

Folgende Aktionen sollen möglich sein:

5.1.6.6.1 Aufzeichnung visualisieren

Der Anwender erstellt mit Hilfe eines Konfigurations-Wizards eine graphische Darstellung für ausgewählte Daten aus einer Aufzeichnung.

5.1.6.6.2 Diagramm konfigurieren

Der Anwender ändert die Einstellungen eines bereits existierenden Diagramms.

5.1.6.6.3 Diagramm exportieren

Der Benutzer exportiert ein erstelltes Diagramm in einem ausgewählten Grafikformat.

5.1.6.6.4 Diagramm drucken

Der Benutzer druckt ein erstelltes Diagramm auf dem Drucker.

5.1.6.6.5 Diagramm speichern

Der Benutzer speichert die Aufzeichnung zusammen mit den Eigenschaften des Diagramms ab.

5.1.7 Spezifizierte Funktionalitäten für die Benutzergruppe „Inbetriebnehmer/Entwickler“

Nachfolgend werden die Funktionalitäten spezifiziert, die für die Benutzergruppe „Inbetriebnehmer/ Entwickler“ vorgesehen sind. Die Abbildung 5-4 zeigt die Bedienfunktionen, die für die Benutzergruppe „Inbetriebnehmer/ Entwickler“ vorgesehen sind.

Funktionsgruppe Antrieb sichern

Die Funktionsgruppe Antrieb sichern ist bereits in Kapitel 5.1.6.5 beschrieben. Zusätzlich zu den dort genannten Funktionen kann der Inbetriebnehmer eine Sicherung auf dem BAUDIS NET-Server löschen.

5.1.7.1.1 Sicherung löschen

Der Benutzer löscht eine auf dem BAUDIS NET Server existierende Sicherung.

5.1.7.2 Funktionsgruppe Softwareupdate

Die Funktionsgruppe Softwareupdate ermöglicht das Installieren oder Aktualisieren der Reglerfirmware des G-Drives. Alle durchgeführten Aktionen in dieser Funktionsgruppe werden in einem Logfile erfasst. Voraussetzung für die Installation oder Aktualisierung der Firmware ist, dass die ausgewählten Regler eine eindeutige Reglerkennung besitzen. Folgende Aktionen müssen möglich sein:

5.1.7.2.1 Antriebe auswählen

Der Benutzer wählt die zu aktualisierenden Antriebe aus eine Antriebsliste aus.

5.1.7.2.2 Firmware auswählen

Der Benutzer wählt die Firmware, die auf die zu aktualisierenden Antriebe geladen werden soll aus.

5.1.7.2.3 Durchführen des Softwareupdates

Nach der Anzeige eines Warnhinweises wird das Softwareupdate durchgeführt.

5.1.7.3 Funktionsgruppe Datensatzverwaltung

Die Funktionsgruppe Datensatzverwaltung ermöglicht das Laden und Lesen eines vollständigen Parameterdatensatzes auf bzw. von eine(r) Auswahl von Reglern. Alle durchgeführten Aktionen in dieser Funktionsgruppe werden in einem Logfile erfasst. Folgende Aktionen sollen möglich sein:

5.1.7.3.1 Auswahl der Antriebe

Der Benutzer wählt die betreffenden Antriebe aus einer Antriebsliste aus.

5.1.7.3.2 Auswahl des Datensatzes

Der Benutzer wählt den Datensatz aus, der auf die Antriebsauswahl geladen werden soll.

5.1.7.3.3 Durchführen des Datensatz-Downloads

Der Benutzer lädt den ausgewählten Datensatz auf die Antriebsauswahl.

5.1.7.3.4 Durchführen des Datensatz-Uploads

Der Benutzer lädt einen bestehenden Datensatz von einem Regler.

5.1.7.4 Funktionsgruppe Events konfigurieren (nur G-Drive)

Die Funktionsgruppe „Events konfigurieren“ bietet die Möglichkeit beliebige Ereignisse in der Ereignisanzeige und im Logbuch aufzuzeichnen. Der im Kommunikations-PC des G-Drives vorhandene Event-Broker wird mit Hilfe der unter genannten Funktionen konfiguriert, so dass er die gewünschten Parameterkombinationen auf das Eintreten des konfigurierten Ereignisses überwacht. Beim Eintreten des Ereignisses wird es an den BAUDIS NET Server geschickt und dort in der Ereignisanzeige angezeigt. Folgende Aktionen sollen möglich sein:

5.1.7.4.1 Antriebe auswählen

Der Benutzer wählt die Antrieb aus, für die ein Event konfiguriert oder gelöscht werden soll.

5.1.7.4.2 Event konfigurieren

Der Benutzer konfiguriert ein Ereignis mit Hilfe eines Konfigurations-Wizards

5.1.7.4.3 Event löschen

Der Benutzer löscht einen Event aus eine Liste mit laufenden Events. Standardmäßig vorhandene Events, wie z.B. Fehler können nicht gelöscht werden.

5.1.7.4.4 Eventkonfiguration zum Antrieb schicken

Der Benutzer schickt den konfigurierten Event zu einer Antriebsauswahl und aktiviert ihn.

5.1.7.5 Funktionsgruppe Skripte

Die Funktionsgruppe „Skripte“ bietet die Möglichkeit komplexe Diagnosefunktionen durchzuführen. Um komplexe Abfragen von Parametern zu realisieren können PERL-Skripte geschrieben werden, die auf den entsprechenden Kommunikations-PC geschickt werden und dort ausgeführt werden. Folgende Aktionen sollen möglich sein:

5.1.7.5.1 Laden des Skripts auf den Server

Der Benutzer lädt das Skript von dem Kommunikations-PC eines Antriebs auf den Server.

5.1.7.5.2 Laden des Skripts zum Antrieb

Der Benutzer lädt ein aus einer Liste ausgewähltes Skript auf einen ausgewählten Antrieb.

5.1.7.5.3 Skript editieren

Der Benutzer editiert ein Skript.

5.1.7.5.4 Ausführen des Skripts auf dem Kommunikations-PC

Der Benutzer startet das Skript auf einem Antrieb.

5.2 Bedienoberfläche zur Anlagenkonfiguration

Für die Konfiguration und Einrichtung des BAUDIS NET Application-Servers soll eine spezielle Bedienoberfläche zur Verfügung gestellt werden, die auf die Benutzung von geschultem Personal zugeschnitten ist. Sie enthält komplexe Funktionalitäten, die die Konfiguration einer großen Anlage mit zahlreichen Antriebssystemen wesentlich schneller und einfacher als bisher ermöglicht. Für das Arbeiten mit dieser Benutzeroberfläche ist keine Verbindung zum Application Server notwendig, so dass die Konfigurationsarbeiten vom Inbetriebnahmepersonal auch offline durchgeführt werden können. Die offline erzeugten Daten können dann nach Abschluss der Arbeiten via ftp auf den Application Server geladen werden, sofern eine Kommunikationsverbindung existiert.

5.2.1 Zielgruppe

Geschultes Personal, das mit der Inbetriebnahme einer Anlage betraut ist.

5.2.2 Mehrsprachigkeit

Die Benutzeroberfläche zur Anlagenkonfiguration soll in einer deutschen und englischen Version zur Verfügung stehen.

5.2.3 Implementierung

Nachdem die Benutzeroberfläche zur Inbetriebnahmeunterstützung nur einem begrenzten Benutzerkreis zur Verfügung gestellt werden soll und sie komplexe Funktionen beinhalten soll, ist eine Implementierung als Java-Oberfläche sinnvoll. Hierdurch wird es im Unterschied zur webbasierten Oberfläche auch möglich bestimmte Arbeiten durchzuführen, ohne dass eine Verbindung zum Application Server besteht.

5.2.4 Spezifizierte Funktionalitäten

Die nachfolgend spezifizierten Funktionen sind in Funktionsgruppen und Funktionen gruppiert.

5.2.4.1 Funktionsgruppe Datensatzerstellung

In der Funktionsgruppe Datensatzerstellung kann der Anwender für eine bestimmte Softwareversion einen Parameterdatensatz erstellen und editieren. Der Parameterdatensatz setzt sich aus einer Reihe von Modulen zusammen, die aus der Standarddatenbank entnommen werden können. Folgende Aktionen müssen möglich sein:

- Auswahl der Standard-Parameterdatenbank auf der Festplatte des Anwenders
- Auswahl des Reglers für den der Parameterdatensatz erstellt werden soll
- Auswählen der Module aus der Standarddatenbank durch Drag and Drop aus einem Drivebrowser, der den Inhalt der Standarddatenbank darstellt.
- Editieren der frei änderbaren Parameter, Ansicht der nicht änderbaren Parameter
- Speichern des Parameterdatensatzes in einer temporären Datenbank. Hierbei soll der Datensatz in einem für die Serverkomponente „Anlagenkonfiguration“ geeigneten Format abgelegt werden.
- Übertragen des erstellten Datensatzes auf den Application-Server

5.2.4.2 Funktionsgruppe Anlagenkonfiguration

Die Funktionsgruppe Anlagenkonfiguration ermöglicht das Zusammenstellen einer Anlage aus verschiedenen Komponenten und deren graphische Repräsentation in einem Anlagen-

bild. Das erstellte Anlagenbild wird, zusammen mit der zugehörigen Datenbank nach Abschluss der Arbeiten via ftp auf den BAUDIS NET Application Server geladen.
Folgende Aktionen müssen möglich sein:

- Laden der Komponentendatenbank von der Festplatte des Anwenders
- Auswahl und Konfiguration der Komponente
- Platzieren der Komponente auf einem Anlagenbild
- Speichern der Anlagenkonfiguration in einer temporären Datenbank auf der Festplatte des Anwenders
- Hintergrundbild laden und löschen
- Übertragen der temporären Datenbank in die Datenbank der Serverkomponente „Anlagenkonfiguration“
- Laden der Anlagenkonfiguration aus der BAUDIS NET Datenbank in eine temporäre Datenbank auf der Festplatte des Anwenders. (z.B. für die Durchführung von Änderungen)

6 Spezifikation des Application-Servers

Das vorliegende Kapitel gibt einen Überblick über die Architektur des Application Servers von BAUDIS NET. Nach einem Systemüberblick in Kapitel 6.1 wird versucht, die Vielzahl der benötigten Funktionen in Komponenten zu gruppieren. Im Kapitel 6.2 wird der Funktionsumfang dieser Komponenten erläutert.

Das Kapitel 6.3 beinhaltet den objektorientierten Entwurf des BAUDIS NET Application Servers. Hier wird es sich je nach Ergebnis der objektorientierten Modellierung zeigen, ob die in Kapitel 6.2 gebildeten Komponenten den Komponenten der Implementierung entsprechen.

6.1 Systemüberblick

Aufbauend auf Abbildung 4-1 zeigt die folgende Abbildung eine detailliertere Struktur von BAUDIS NET.

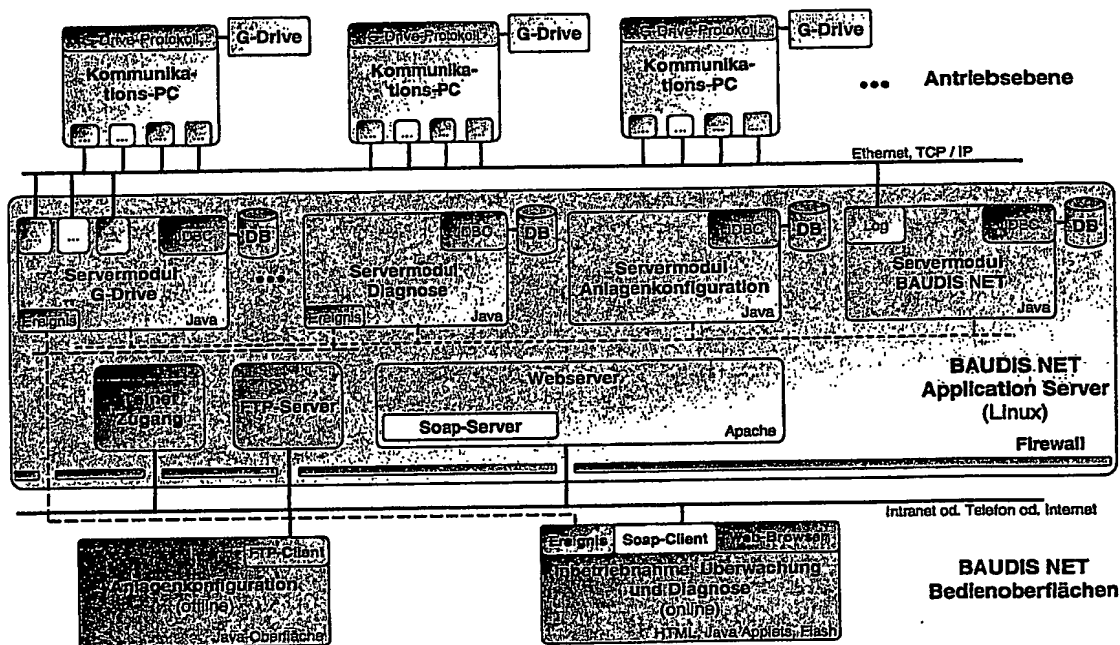


Abbildung 6-1: Detaillierte Struktur von BAUDIS NET

Im wesentlichen besteht BAUDIS NET aus drei Ebenen:

BAUDIS NET Bedienoberflächen:

Alle Funktionalitäten von BAUDIS NET können über die bereits in Kapitel 5 spezifizierten Bedienoberflächen bedient werden. Für den Benutzer von BAUDIS NET soll es keinen Unterschied machen, ob er sich im lokalen Netz an der Anlage befindet oder über das Internet oder eine Telefon-Einwahlverbindung mit dem Application-Server verbunden ist.

BAUDIS NET Application Server

Der BAUDIS NET Application Server stellt den Kern der gesamten Anwendung dar. Seine Funktionalität ist in verschiedene Komponenten unterteilt. Jede Komponente ist in sich abgeschlossen und stellt seine Funktionalität der webbasierten Bedienoberfläche oder anderen Serverkomponenten zur Verfügung. Alle für die Funktion der Komponente notwendigen Daten werden in der angeschlossenen Datenbank gespeichert. Um die Kapselung und Konsistenz dieser Daten zu gewährleisten kann auf diese Datenbestände nur über die von der Serverkomponente zur Verfügung gestellten Funktionen zugegriffen werden. Dadurch wird auch

sichergestellt, dass eine Änderung an der Datenbankstruktur einer Serverkomponente nicht unbedingt Änderungen an anderen Serverkomponenten nach sich zieht.

Um die Funktionalitäten der Serverkomponenten den Bedienoberflächen zur Verfügung zu stellen soll eine geeignete Infrastruktur geschaffen werden. Für die Kommunikation mit der Weboberfläche zur Inbetriebnahme, Überwachung und Diagnose soll ein Apache Webserver eingesetzt werden. Dieser stellt HTML-Seiten zur Verfügung, in die Java-Applets oder Flash-Module eingebettet sind. Die auf der Oberfläche anzuzeigenden Daten werden mit Hilfe von Soap (Simple Object Application-Protokoll) zu den entsprechenden Modulen übertragen. Die Benutzeroberfläche ruft z.B. eine Funktion der Serverkomponente G-Drive auf. Die zu übergebenden Parameter und die Bezeichnung des Funktion wird mit Hilfe des Soap Protokolls auf das Intra- oder Internet geschickt. Um die Transparenz gängiger Firewalls zu gewährleisten, wird der Funktionsaufruf in Form eines HTTP-Telegramms verschickt. Ein Webserver auf der Seite des Application Servers empfängt die HTTP-Telegramme mit einer Soap-Inhalt und leitet sie an den Soap-Server weiter. Der Soap-Server decodiert die Anfrage und ruft die gewünschte Funktion aus der Serverkomponente G-Drive auf. Die Funktion wird ausgeführt und die Rückgabewerte wiederum in das Soap Protokoll umgesetzt und als HTTP-Telegramm an die Oberfläche geschickt.

Eine wesentliche Eigenschaft eines Diagnose- und Überwachungssystems ist, dass der Benutzer über an der Anlage auftretende Ereignisse zeitnah informiert wird. Dies stellt für die oben gezeigte Architektur eine Schwierigkeit dar, da sowohl für die Kommunikation über Soap als auch für die HTML-Seiten eine ereignisbasierte Benachrichtigung nicht vorgesehen ist. Als Abhilfe müssen an der Anlage auftretende Ereignisse, wie z.B. das Auftreten eines Fehlers oder das Update eines Parameterwertes in einer Oberfläche, über einen Event-Kanal den Benutzeroberflächen mitgeteilt werden. Nachdem diese Verbindung über TCP/IP läuft ist für die Verbindung über das Internet ein offener Port am Firewall notwendig.

Antriebsebene:

Die Antriebsebene stellt dem Application Server die Daten von den Reglern zur Verfügung. In Abbildung 6-1 sind nur G-Drives mit daran angeschlossenen Kommunikations-PC dargestellt, es sollen jedoch zukünftig auch andere Reglertypen wie z.B. Regler aus der b-maXX Familie unterstützt werden können.

6.2 Die Serverkomponenten

Wie bereits in Kapitel 6.1 beschrieben besteht der Application Server aus verschiedenen gekapselten Serverkomponenten die ihre Funktionalitäten der Benutzeroberfläche zur Verfügung stellen. Durch die komponentenbasierte Struktur soll sichergestellt werden, dass der Funktionsumfang von BAUDIS NET erweitert werden kann. Die Serverkomponenten sind als Java-Module realisiert.

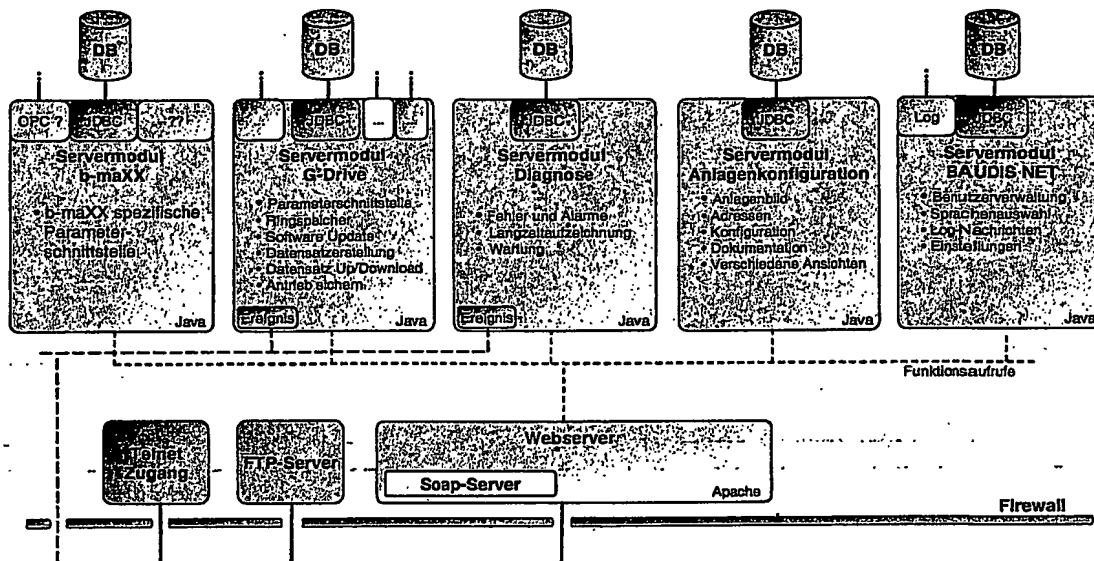


Abbildung 6-2: Verschiedene Serverkomponenten

Die Abbildung 6-2 zeigt die verschiedenen Serverkomponenten von BAUDIS NET. Die folgenden Kapitel spezifizieren die einzelnen Serverkomponenten:

6.2.1 Serverkomponente BAUDIS NET

Die Serverkomponente BAUDIS NET enthält die für den Betrieb von BAUDIS NET notwendigen Funktionalitäten. Es enthält eine Reihe von Funktionen die im folgenden kurz erklärt werden:

Benutzerverwaltung

Alle Funktionalitäten in BAUDIS NET sind gegen unautorisierten Zugriff geschützt. Jeder Benutzer hat eine Benutzerkennung und gehört zu einer Benutzergruppe, die ihm ein Rechteprofil für den Zugriff auf die Funktionalitäten der Serverkomponenten ermöglicht. Diese Informationen werden in der Benutzerverwaltung konfiguriert und abgespeichert. Jede Funktion in den Serverkomponenten hat eine eindeutige Kennung. Will ein Benutzer auf eine Funktion zugreifen, wird zunächst an der Serverkomponente Benutzerverwaltung nachgefragt, ob der Benutzer die entsprechenden Rechte für das Ausführen dieser Funktion besitzt. Die Datenbank, in der die Benutzerdaten abgelegt werden sollen, durch Verschlüsselung vor unberechtigtem Zugriff geschützt werden. Folgenden Funktionen sollen zur Verfügung gestellt werden:

- Benutzer einrichten
- Benutzer löschen
- Rechte des Benutzers editieren
- Autorisierung prüfen (intern)

Sprachenauswahl

BAUDIS NET soll mehrere Sprachen unterstützen. Die Auswahl der Sprache geschieht vor dem Anmelden an der Benutzeroberfläche. Für die Bedienoberfläche zur Anlagenkonfiguration ist nur ein eingeschränkter Sprachumfang vorgesehen (z.B. Deutsch und Englisch). Fehlertexte und andere Meldungen werden in der entsprechenden Sprache an die Oberfläche übertragen. Für die Weboberfläche werden für jede zu unterstützende Sprache entsprechende HTML-Seiten zur Verfügung gestellt.

Log-Nachrichten

Alle Vorgänge die von einer Benutzeroberfläche beauftragt werden, sollen in einem Log-File aufgezeichnet werden sein. Hierbei wird jedem Vorgang eine eindeutige Kennung zugeordnet. Bei Aufruf wird ein entsprechender Eintrag in die Log-Datenbank erzeugt. Bei erfolgreichem Abschluss wird dieser Eintrag abgeschlossen.

Wesentliche Vorgänge auf den Kommunikations-PCs und anderen Reglern können ebenfalls in eine dafür vorgesehene Datenbank geschrieben werden. Sie sollen mit derselben eindeutigen Kennung versehen werden, so dass eine alle Aktionen durchgängig einander zugeordnet werden können. Folgenden Funktionen sollen zur Verfügung gestellt werden:

- Konfigurieren des Loggings
- Einstellen der Detailtiefe
- Komprimieren und Sichern der Log-Datenbank

Einstellungen

Weitere, für den Betrieb und die Konfiguration des Application Servers notwendige Einstellungen sollen hier zentral vorgenommen werden können.

6.2.2 Serverkomponente Anlagenkonfiguration

Die Serverkomponente Anlagenkonfiguration enthält alle notwendigen Daten über die Konfiguration der Anlage. Diese beinhaltet Daten über die in der Anlage vorhandenen Komponenten, Gruppierung der Komponenten, Adressen etc. Es sollen Funktionalitäten zur Verfügung gestellt werden, die die Darstellung der Anlagenkonfiguration in Form verschiedener Übersichtsbilder erlauben. (Siehe dazu auch Kapitel 5.1.5.5) Weiterhin sollen alle Dokumentationen zu den in der Anlage enthaltenen Daten zur Verfügung gestellt werden.

Beim Entwurf der Serverkomponente Anlagenkonfiguration ist darauf zu achten, dass mit Hilfe der Funktionalitäten dieser Komponente beliebige Anlagen im Bereich der Antriebstechnik beschrieben werden können.

6.2.3 Serverkomponente G-Drive

Die Serverkomponente G-Drive enthält alle G-Drive spezifischen Funktionalitäten. Hierzu gehören das Lesen und Schreiben von Parametern, die Bedienung des Ringspeichers sowie Funktionen für das Softwareupdate und den Up-/Download von Parameterdatensätzen.

6.2.4 Serverkomponente Diagnose

Die Serverkomponente Diagnose beinhaltet die Diagnose-Funktionalitäten von BAUDIS NET. Hierzu gehört die Aufzeichnung von Ereignissen (z.B. Fehlern) sowie die Aufzeichnung von beliebigen Daten. Die zu erfassenden Ereignisse sollen konfigurierbar sein und unabhängig davon ob sie von einer Oberfläche angefordert werden in Form von Logbüchern in der Datenbank abgelegt werden.

Um die Allgemeingültigkeit der Komponente Diagnose zu erhalten, sollen hier keine reglerspezifischen Funktionalitäten enthalten sein. Für reglerspezifische Funktionalitäten wie z.B. das Lesen und Schreiben von Parametern soll auf die entsprechende Serverkomponente zugegriffen werden.

6.2.5 Serverkomponente b-maXX

Die Serverkomponente b-maXX enthält alle b-maXX spezifischen Funktionalitäten. Eine weitergehende Spezifikation dieser Serverkomponente findet in einem separaten Dokument statt und ist nicht Bestandteil des vorliegenden Entwicklungsprojekts.

einem oder zwei Geber, einem Leistungsteil, einem Motor und einem Regler. Vom Regler gibt es verschiedene Typen. Er kann eine Firmware und einen Datensatz enthalten.

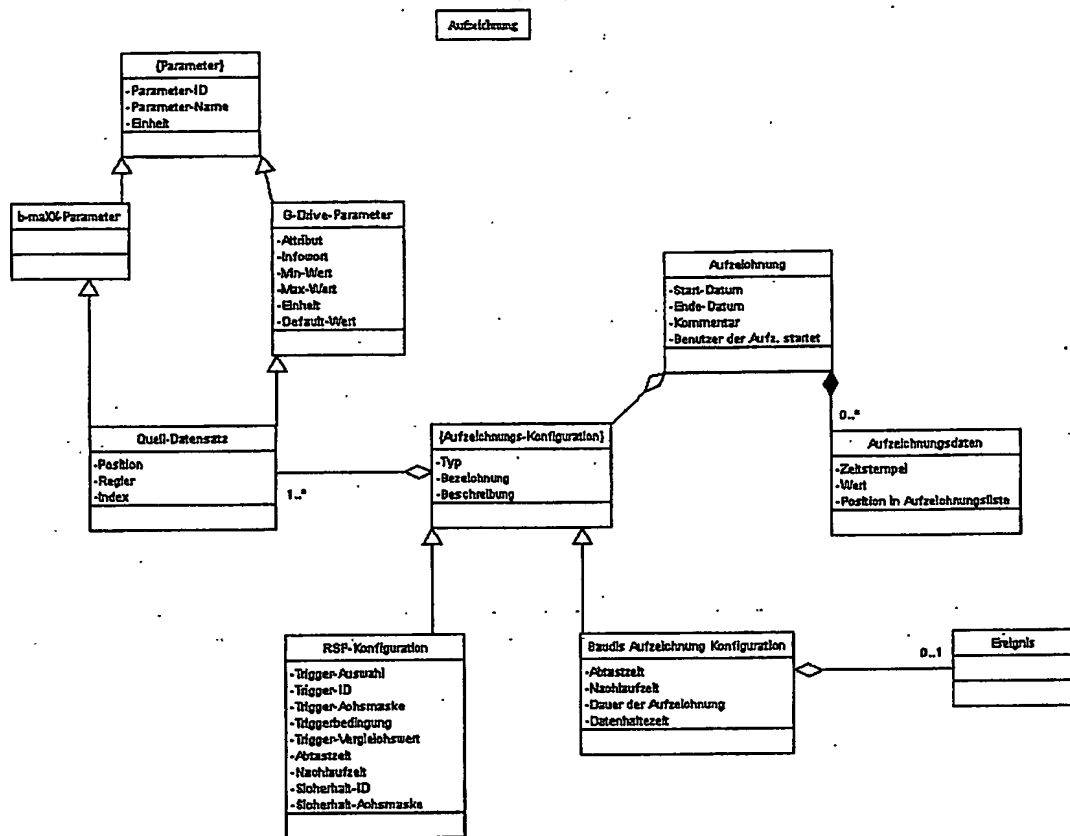


Abbildung 6-4: UML-Klassendiagramm für eine Aufzeichnung

Die Abbildung 6-4 zeigt ein Klassendiagramm für eine Aufzeichnung. Teil einer Aufzeichnung ist eine Aufzeichnungskonfiguration sowie Aufzeichnungsdaten, sofern die Aufzeichnung bereits einmal gestartet wurde. Die Klasse Aufzeichnungskonfiguration ist abstrakt. Von ihr erben zwei Klassen: Die Klasse Ringspeicher-Konfiguration sowie die Klasse Baudis Aufzeichnungskonfiguration. Einem Objekt vom Typ Baudis Aufzeichnungskonfiguration kann mit einem Ereignis verknüpft werden, das z.B. eine laufende Aufzeichnung nach dem Auftreten des Ereignisses beendet.

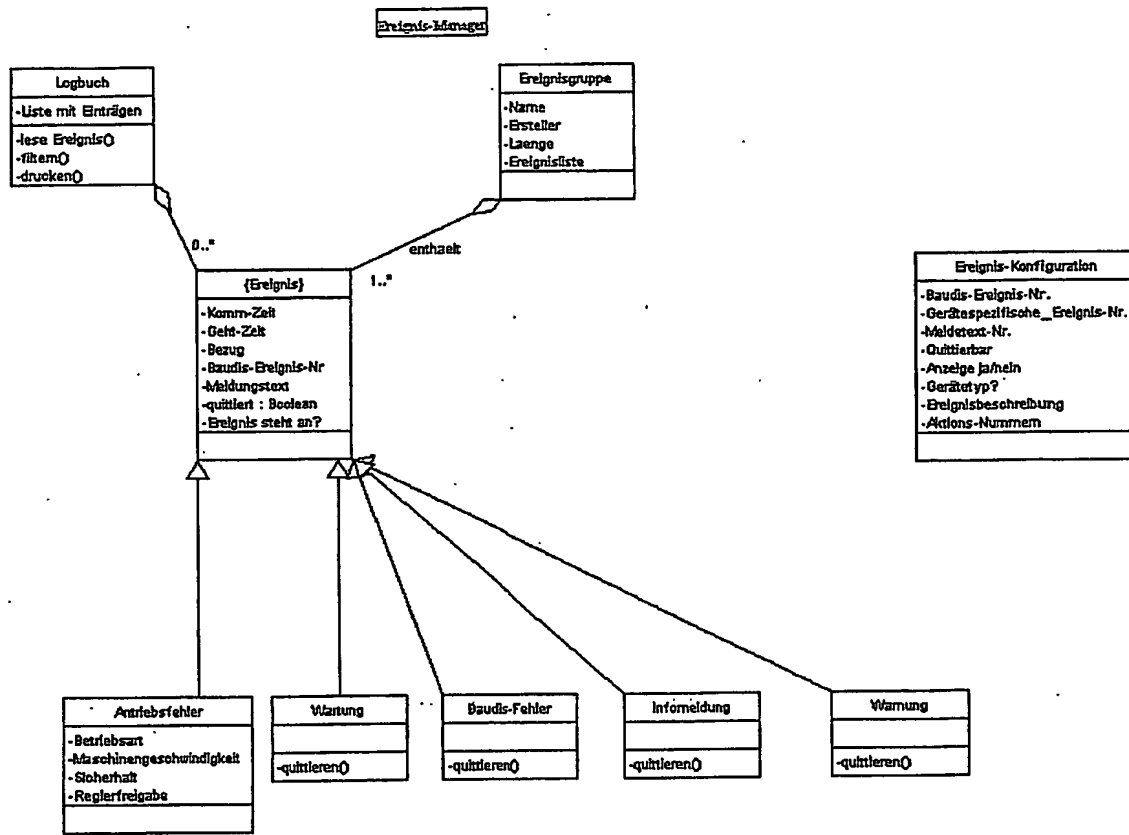


Abbildung 6-5: UML-Klassendiagramm für den Ereignismanager

Die Abbildung 6-5 zeigt Ausschnitte aus einem Klassendiagramm für den Ereignismanager in BAUDIS NET. Der Ereignismanager verwaltet alle im Antriebssystem möglichen Ereignisse (z.B. Fehler, Wartungsmeldungen, Infos, Warnungen...). Wird ein Ereignis von einer beliebigen Komponente an den Ereignismanager gemeldet wird es vom Ereignismanager weiter verarbeitet. Z.B. kann mit einem Ereignis eine Aktion verknüpft werden, wie z.B. die Anzeige in der Ereignisanzeige der Bedienoberfläche oder mit dem Verschicken einer e-mail.

Ebene 2: Darstellungs-Logik

Auf der zweiten Ebene werden Klassen spezifiziert, welche die auf den Bedienoberflächen dargestellten Funktionen realisieren. Dafür werden die Klassen der Modellebene benutzt.



Spezifikation des Kommunikations-PC (Software)

Stand: 17.07.2002

Thomas Tschaftary, Harold Meis

Inhaltsverzeichnis

1	Systemüberblick	3
2	Spezifikation des Request-Brokers „Parameter“	6
2.1.1	Spezifizierte Funktionalitäten	6
2.1.2	Funktionsweise	6
2.1.3	Schnittstellen-Spezifikation Request-Broker „Parameter“ – Client.....	7
2.1.4	Schnittstellen-Spezifikation Request-Broker „Parameter“ – G-Drive.....	11
3	Spezifikation des Request-Brokers „Fehler und Events“	12
3.1.1	Spezifizierte Funktionalitäten	12
3.1.2	Funktionsweise	12
3.1.3	Schnittstellen-Spezifikation Request-Broker „Fehler und Events“ – Client.....	13
3.1.4	Schnittstellen-Spezifikation Request-Broker „Fehler und Events“ – Request-Broker „Parameter“	15
4	Spezifikation des Request-Brokers „Parameter-Bedarfsdaten“	16
5	Spezifikation des Request-Brokers „Zyklische Sollwerte“	17
6	Spezifikation des Request Brokers „Softwaredownload“	18
7	Spezifikation des Connection-Managers	19
8	Weitere Dienste auf dem Kommunikations-PC.....	20
8.1	Spezifikation des Logging Servers	20
8.2	Spezifikation der Skriptunterstützung	20
8.3	Zeitsynchronisation, FTP-Server, Telnet.....	20

Änderungsgeschichte

Version	Datum	Ersteller	Art der Änderung	Freigabe
1.0	17.07.02	Tschafary, Meis	Ersterstellung	

1 Systemüberblick

Der Kommunikations-PC übernimmt die Kommunikationsaufgaben zwischen dem Regler und der Außenwelt. In der vorliegenden Spezifikation wird die Softwarestruktur für die Kommunikation zum G-Drive spezifiziert.

Jedes Gerät, das mit dem G-Drive kommunizieren soll muss ihn über eine geeignete Software-Schnittstelle auf dem Kommunikations-PC ansprechen. Mit Hilfe des Kommunikations-PCs kann beinahe jede Software-Schnittstelle, die auf Ethernet basiert, realisiert werden.

Das folgende Kapitel gibt einen Überblick über die Softwarearchitektur auf dem Kommunikations-PC. Eine Einordnung in das Gesamtkonzept von BAUDIS NET ist der Spezifikation von BAUDIS NET, Kapitel 4 zu entnehmen. Die Abbildung 1-1 zeigt die Softwarestruktur auf dem Kommunikations-PC.

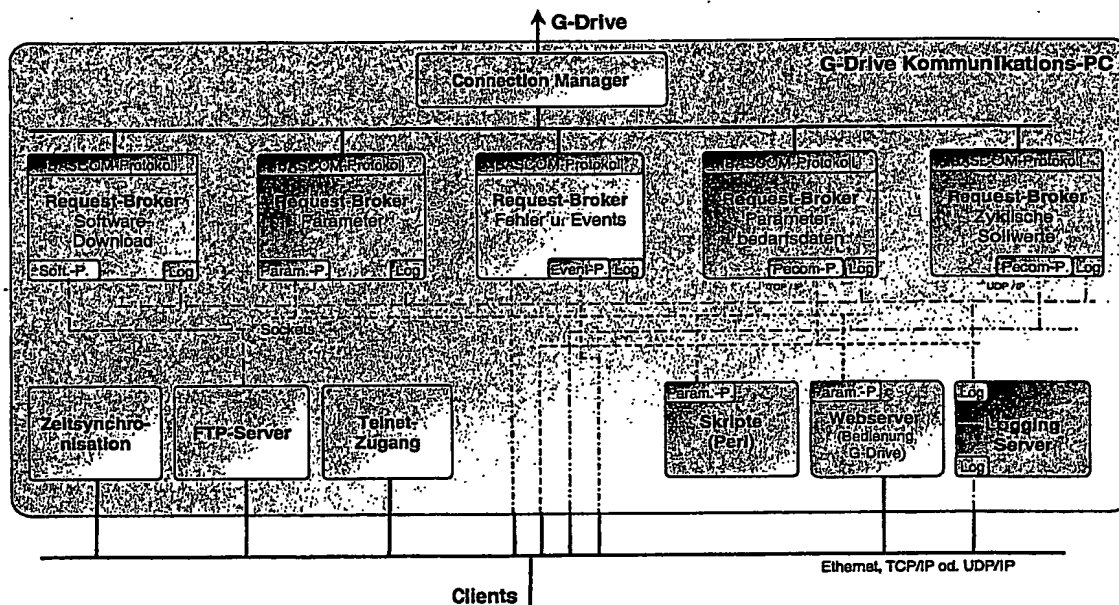


Abbildung 1-1: Softwarestruktur auf dem Kommunikations-PC

Jede Funktionalität, die vom G-Drive der Außenwelt zur Verfügung gestellt werden soll, ist in einem Softwaremodul (Request-Broker) realisiert. Z.B. stellt der Request-Broker „Parameter“ eine Parameterschnittstelle zur Verfügung über die beliebige Parameter des G-Drives gelesen und geschrieben werden können. Der Request-Broker „Fehler und Events“ stellt beliebige Ereignisse und Fehler nach außen dar. Die beiden Request-Broker „Parameter-Bedarfsdaten“ und „Zyklische Sollwerte“ erledigen die Kommunikation mit der Steuerung. Sie sind nur auf den Reglern vorhanden die den Master im Sercos-Ring bilden und somit mit der Steuerung kommunizieren müssen. Der Request-Broker „Software-Download“ stellt Funktionen für einen automatischen Download der Regler-Firmware bereit.

Für jede an einem Request-Broker eingehende Anfrage wird ein eigener Prozess erzeugt, der die Anfrage bearbeitet. So wird sichergestellt, dass ein Request-Broker viele Anfragen gleichzeitig bedienen kann.

Im wesentlichen kommuniziert der Kommunikations-PC des G-Drive mit der Steuerung und dem BAUDIS NET Application Server. Hierbei muss sichergestellt werden, dass die Nachrichten von und zur Steuerung in jedem Fall ohne unnötigen Zeitverzug am G-Drive verarbeitet werden, da sie für den Betrieb der Anlage von wesentlicher Bedeutung sind. Nachdem die Anfragen auf dem G-Drive nur sequentiell verarbeitet werden können, muss die Möglichkeit bestehen, Anfragen von der Steuerung vorrangig zu bearbeiten. Dies soll

durch Vergabe von Prioritäten für die Anfragen sichergestellt werden. Jede Anfrage an die den Connection-Manager, der die Schnittstelle zum G-Drive verwaltet, ist mit einer von 5 Prioritätsstufen versehen. Anfragen mit der höchsten Priorität werden vom Connection-Manager vor allen weiteren anstehenden Anfragen an den G-Drive geschickt. Anfragen mit niedriger Priorität werden immer nach allen anderen anstehenden Aufträgen behandelt. Hierdurch kann sichergestellt werden, dass ein Auftrag mit höchster Priorität –z.B. von der Steuerung- in jedem Fall als nächste Anfrage auf dem G-Drive bearbeitet wird.

Neben den Request-Brokern gibt es weitere optionale Softwaremodule auf dem Kommunikations-PC: Ein Logging-Server empfängt Log- und Debugnachrichten von den Request-Brokern und stellt sie der Außenwelt zur Verfügung. Ein Webserver bietet eine einfache Weboberfläche zur Bedienung und Konfiguration. Ein FTP-Server und eine Client zur Zeitsynchronisation sind weitere Komponenten auf dem Kommunikations-PC. Mit einem Perl-Interpreter können beliebige Skripte mit Diagnose- oder Steuerungsfunktionalitäten ausgeführt werden.

2 Spezifikation des Request-Brokers „Parameter“

Das Kapitel 2 spezifiziert die Funktionalitäten des Request-Brokers „Parameter“ und die dazugehörigen Kommunikations-Protokolle.

2.1 Spezifizierte Funktionalitäten

Folgende Funktionen sollen im Request-Broker „Parameter“ verfügbar sein:

Lesen von Parametern

Der Request-Broker Parameter soll eine Gruppe von beliebigen Parametern von G-Drive lesen können. Hierbei soll neben dem einmaligen Lesen ein zyklisches Lesen von Parametern möglich sein. Das Intervall zwischen den Lesevorgängen und die Anzahl der Lesevorgänge sollen vom Client einstellbar sein.

Um die Kommunikation mit dem G-Drive zu verringern soll der Request-Broker „Parameter“ alle bereits gelesenen Parameterinformationen (z.B. Attribut, Name, Einheit) in einem Cache ablegen, so dass sie beim erneuter Anforderung des Parameters nicht erneut vom Regler gelesen werden müssen. Dynamische Parameter, die ihre Eigenschaften zur Laufzeit verändern können sollen nicht in den Cache aufgenommen werden. Sie müssen jedes Mal vollständig gelesen werden.

Schreiben von Parametern

Der Request-Broker Parameter soll eine Gruppe von beliebigen Parametern auf den G-Drive schreiben können. Hierzu müssen die zu schreibenden Parameter in das benötigte Format gewandelt werden, was vorab ein Lesen der Parameterinformationen erfordert, sofern sie nicht im Cache verfügbar sind.

2.2 Funktionsweise

Abbildung 2-1 gibt einen Überblick über den groben Ablauf beim Verarbeiten einer Anfrage im Request-Broker „Parameter“.

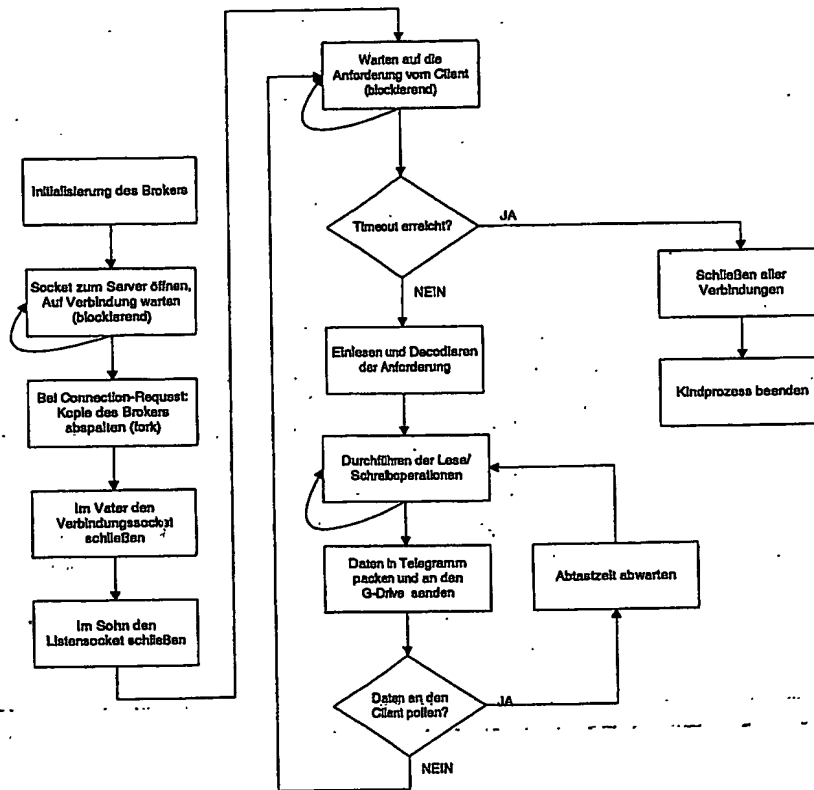


Abbildung 2-1: Flussdiagramm Request-Broker „Parameter“

Nach der Initialisierung des Brokers wird ein Socket (Listensocket) an einem konfigurierbaren Port geöffnet und auf eine Verbindungsanfrage eines Clients gewartet. Verbindet sich ein Client mit diesem Listensocket wird eine Kopie des Brokerprozesses erzeugt, die die Anfrage des Clients bearbeitet. Der ursprüngliche Prozess (Vaterprozess) wartet weiterhin an seinem Listensocket auf sich verbindende Clients. So ist sichergestellt, dass sich viele Clients gleichzeitig vom Broker bearbeitet werden können.

Nach der Prozeßabspaltung schließt der Vaterprozess sein Verbindungssocket und der Kindprozess sein Listensocket. Danach ist der Kindprozess bereit zum empfangen der Anfrage vom Client. Bei Eintreffen der Anfrage wird diese vom Socket gelesen und decodiert. Je nach Typ der Anfrage wird diese entsprechend ausgeführt.

Nach Verarbeitung der Anfrage vom Client wartet der Kindprozess eine bestimmte Zeitspanne auf neue Anfragen vom Client. Trifft innerhalb der vordefinierten Zeit keine neue Anfrage vom Client ein oder tritt während der Bearbeitung des Anfrage ein Fehler auf, wird die Verbindung beendet.

2.3 Schnittstellen-Spezifikation Request-Broker „Parameter“ – Client

Im folgenden Kapitel wird die Kommunikations-Schnittstelle zwischen dem Request-Broker „Parameter“ und dem Client spezifiziert. Es soll hier ein einfaches und schnelles proprietäres Protokoll zum Einsatz kommen.

In den folgenden Abschnitten werden die benötigten Telegramme spezifiziert. Jeder empfangene Protokollstring wird beim Decodieren soweit möglich auf Gültigkeit der enthaltenen Daten überprüft. Bei Auftreten eines Fehlers wird dieser in einer speziellen Fehler/Bestätigungsnachricht an den Client zurückgeschickt. Nach erfolgreicher Bearbeitung der Anfrage wird eine Bestätigungsnachricht an den Client versendet.

Die Verbindung zwischen Server und dem Request-Broker kann durch Schließen des Sockets von beiden Seiten jederzeit beendet werden.

Alle Daten werden als String übertragen. Das hat den Vorteil, dass das Protokoll auch von „primitiven“ Clients (z.B. Skripte) oder sogar „von Hand“ benutzt werden kann. Zwischen den Elementen des Telegramms befinden sich Trennzeichen. Am Telegrammanfang und am Telegrammende stehen jeweils verschiedene Kennungen.

Client → Request-Broker: Daten anfordern:

Die nachfolgende Tabelle 2-1 spezifiziert die Anfrage des Clients an den Request-Broker „Parameter“ zur Übermittlung von Daten. (Die grau hinterlegten Bereiche des Telegramms sind in allen Telegrammen vorhanden).

Datum	Stringlänge	Gültigkeit	Beschreibung
BEOT			Beginn des Telegramms
Kennung	20	Zeichen	Kennung des Servers/ Brokers (für Logging-Zwecke) Stringlänge: max_serverkennung_length
Zeitstempel	30	Zeichen	Zeit zu der das Telegramm abgeschickt wurde Stringlänge: max_zeitstempel_length
Msg-Typ	Keine Begrenzung (k.B.)	0 ≤ x ≤ 3	Gibt die Art der Message an: 1: Parameterabfrage (im folgenden dargestellt) 2: Parameter schreiben Werte größer 2 nur gültig für RB → Client Gültigkeit: max_msg_typ
Priorität	k.B.	0 < x ≤ 10	Priorität mit der die Anfrage behandelt werden soll Gültigkeit: max_prio
Pollingstatus	k.B.	0 < x ≤ 3	Gibt an, wie oft die Daten gesendet werden: 1: Übertragung gemäß Polling Anzahl 2: Übertragung unbegrenzt Gültigkeit: max_polling_status
Polling Anzahl	k.B.	x ≥ 0	Gibt an, wie oft die Datensätze maximal gelesen und gesendet werden (nur gültig bei Pollingstatus 1)
Polling Intervall	k.B.	x ≥ 0	Zeitintervall nach dem die Daten erneut gelesen und gesendet werden: 0: so schnell wie möglich (x): Intervall in Millisekunden
Anzahl P-IDs	k.B.	0 < x ≤ 20	Anzahl der geforderten Parameter Gültigkeit max_pid_anzahl
P-ID Nr. 1	k.B.	x > 0	Parameter-ID des ersten geforderten Parameters
P-Index Nr.1	k.B.	x ≥ 0	Parameter-Index zum Parameter Nr.1
P-ID Nr. 2	k.B.	x > 0	Parameter-ID des zweiten geforderten Parameters
P-Index Nr. 2	k.B.	x ≥ 0	Parameter-Index zum Parameter Nr. 2
P-ID Nr. 3	k.B.	x > 0	Parameter-ID des dritten geforderten Parameters
P-Index Nr. 3	k.B.	x ≥ 0	Parameter-Index zum Parameter Nr. 3
....			
P-ID Nr. x	k.B.	x > 0	Parameter-ID des x-ten geforderten Parameters
P-Index Nr. x	k.B.	x ≥ 0	Parameter-Index zum Parameter Nr. x
EOT			Ende des Telegramms

Tabelle 2-1: Kommunikationsprotokoll zum Lesen von Daten

Client → Request-Broker: Daten schreiben

Die nachfolgende Tabelle 2-2 zeigt die Aufforderung des Clients an den Request-Broker Daten zu schreiben.

Datum	Stringlänge	Gültigkeit	Beschreibung
BEOT	Siehe oben	Siehe oben	Beginn des Telegramms
Kennung			Kennung des Servers/Brokers (für Logging-Zwecke)
Zeitstempel			Zeit zu der das Telegramm abgeschickt wurde
Msg-Typ			Gibt die Art der Message an: 1: Parameterabfrage 2: Parameter schreiben (im folgenden dargestellt)
Priorität	Priorität mit der die Anfrage behandelt werden soll
Anzahl P-IDs	Siehe oben	Siehe oben	Anzahl der geforderten Parameter Gültigkeit max_pid_anzahl
P-ID Nr. 1	Siehe oben	Siehe oben	Parameter-ID des ersten geforderten Parameters
P-Index Nr. 1	Siehe oben	Siehe oben	Parameter-Index zum Parameter Nr. 1
Parameterwert	k.B.	Stringl	Wert des zu schreibenden Parameters
P-ID Nr. 2	Siehe oben	Siehe oben	Parameter-ID des ersten geforderten Parameters
P-Index Nr. 2	Siehe oben	Siehe oben	Parameter-Index zum Parameter Nr. 2
Parameterwert	k.B.	Stringl	Wert des zu schreibenden Parameters
P-ID Nr. 3	Siehe oben	Siehe oben	Parameter-ID des ersten geforderten Parameters
P-Index Nr. 3	Siehe oben	Siehe oben	Parameter-Index zum Parameter Nr. 3
Parameterwert	k.B.	Stringl	Wert des zu schreibenden Parameters
...
P-ID Nr. x	Siehe oben	Siehe oben	Parameter-ID des ersten geforderten Parameters
P-Index Nr. x	Siehe oben	Siehe oben	Parameter-Index zum Parameter Nr. x
Parameterwert	30 Zeichen	Stringl	Wert des zu schreibenden Parameters
EOT			Ende des Telegramms

Tabelle 2-2: Kommunikationsprotokoll zum Schreiben von Daten

Request-Broker → Client: Fehlermeldung senden:

Nach Empfang eines Telegramms durch den Broker wird bei Auftreten eines Fehlers eine Fehlermeldung an den Client verschickt. In Fehler steht die Fehlermeldung, die zum Abbruch des Brokerprozesses geführt hat.

Datum	Stringlänge	Gültigkeit	Beschreibung
BEOT			Beginn des Telegramms
Kennung	Siehe oben	Siehe oben	Kennung des Servers/Brokers (für Logging-Zwecke)
Zeitstempel	Siehe oben	Siehe oben	Zeit zu der das Telegramm abgeschickt wurde
Msg-Typ	Siehe oben	Siehe oben	Gibt die Art der Message an: 1: Parameterabfrage (im folgenden dargestellt) 2: Parameter schreiben (siehe Tabelle 2-2) 3: Fehlermeldung
Fehler-Nr.	k.B.	$X > 0$	Eindeutige Fehlernummer
Fehlerstring	300 Zeichen		Beschreibung des Fehlers
EOT			Ende des Telegramms

Tabelle 2-3: Fehlermeldung

Request-Broker → Client: Parameter schreiben - Bestätigung:

Nach dem Schreiben von Parametern durch den Broker wird eine Bestätigung an Client verschickt.

Datum	Stringlänge	Gültigkeit	Beschreibung
BEOT	Siehe oben	Siehe oben	Beginn des Telegramms
Kennung	Siehe oben	Siehe oben	Kennung des Servers/ Brokers (für Logging-Zwecke)
Zeitstempel	Siehe oben	Siehe oben	Zeit zu der das Telegramm abgeschickt wurde
Msg-Typ	Siehe oben	Siehe oben	Gibt die Art der Message an: 1: Parameterabfrage 2: Parameter schreiben 3: Fehlermeldung 4: Parameter schreiben - Bestätigung
Anzahl PIDs	Siehe oben	Siehe oben	Anzahl der PID- Nummern
P-ID Nr. 1	Siehe oben	Siehe oben	Parameter-ID des ersten geschriebenen Parameters
P-Index Nr. 1	Siehe oben	Siehe oben	Parameter-Index zum Parameter Nr. 1
Fehler Nr 1	k.B.	X >= 0	Falls der Parameter nicht geschrieben werden konnte: Fehlernr. vom G-Drive (0 für erfolgreich)
...			
P-ID Nr. 2	Siehe oben	Siehe oben	Parameter-ID des zweiten geschriebenen Parameters
...			
EOT			Ende des Telegramms

Tabelle 2-4: Bestätigungs-Nachricht für das Schreiben von Parametern

Request-Broker → Client Daten senden:

Für die (zyklische) Übertragung der Daten vom Broker zum Server wird das Telegramm aus Tabelle 2-5 verwendet.

Datum	Stringlänge	Gültigkeit	Beschreibung
BEOT	Siehe oben	Siehe oben	Beginn des Telegramms
Kennung	Siehe oben	Siehe oben	Kennung des Servers/Brokers (für Logging-Zwecke)
Zeitstempel	Siehe oben	Siehe oben	Zeit zu der das Telegramm abgeschickt wurde
Msg-Typ	Siehe oben	Siehe oben	Gibt die Art der Message an: 1: Parameterabfrage 2: Parameter schreiben 3: Fehlermeldung 4: Parameter schreiben - Bestätigung 5: Parameter senden (im folgenden dargestellt)
Antwort Typ	Siehe oben	Siehe oben	Typ der Antwort: 1: kurze Antwort (nur PID-Nr, P-Index und Wert, und Fehler gültig) 2: lange Antwort (gesamter Datensatz)
Anzahl P-IDs	Siehe oben	Siehe oben	Anzahl der geforderten Parameter (Gibt es eine Obergrenze x ?)
P-ID Nr. 1	Siehe oben	Siehe oben	Parameter-ID des ersten geforderten Parameters
P-Index Nr. 1	Siehe oben	Siehe oben	Parameter-Index zum Parameter Nr. 1
P-Wert Nr. 1	Siehe oben	Siehe oben	Wert von P-ID Nr. 1
Fehler	Siehe oben	Siehe oben	Falls der Parameter nicht gelesen werden konnte
P-Name Nr. 1	80 Zeichen		Name und Beschreibung des Parameters
P-Attribut Nr.1	Bitmaske Unsigned int		Attribut-Wort: Datenlänge, Datentyp, Anzeigeformat etc.

P-Info Nr. 1	Bitmaske Unsigned int		Info-Wort: Schreibschutz, Speicher-Modus
P-Einheit Nr. 1	20 Zeichen		Einheit des Parameters
Maxwert Nr. 1	30 Zeichen		Maximalwert des gelesenen Parameters
Minwert Nr. 1	30 Zeichen		Minimalwert des gelesenen Parameters
.....			
P-ID Nr. 2			Parameter-ID des zweiten geforderten Parameters
.....			
P-ID Nr. 3			Parameter-ID des dritten geforderten Parameters
....			
P-ID Nr. x			Parameter-ID des x-ten geforderten Parameters
...	
P-ID Nr. x			Parameter-ID des ersten geforderten Parameters
EOT			Ende des Telegramms

Tabelle 2-5: Kommunikationsprotokoll zum Senden von Daten

2.4 Schnittstellen-Spezifikation Request-Broker „Parameter“ – G-Drive

Für die Kommunikation des Request Brokers Parameter mit dem G-Drive soll das BASCOM-Protokoll eingesetzt werden. Das BASCOM-Protokoll ist kompatibel zum PECOM-Protokoll. Es erweitert das PECOM-Protokoll um file-transfer Funktionalitäten.

Eine weitergehende Spezifikation des BASCOM-Protokolls findet im Dokument „Spezifikation des BASCOM-Protokolls“ statt.

3 Spezifikation des Request-Brokers „Fehler und Events“

Das Kapitel 3 spezifiziert die Funktionalitäten des Request-Brokers „Fehler und Events“ und die dazugehörigen Kommunikations-Protokolle.

3.1 Spezifizierte Funktionalitäten

Der Request-Broker „Fehler und Events“ soll am G-Drive auftretende Fehler oder andere konfigurierbare Ereignisse der Außenwelt zur Verfügung stellen. Über die unten spezifizierte Telegramme soll es möglich sein, beliebige Parameter zu überwachen und beim Eintreten eines Ereignisses eine Nachricht an den Client zu verschicken.

Ein Unterschied zum Request-Broker „Parameter“ ist, dass zum Client keine permanente Socket-Verbindung besteht. Nach der Konfiguration des Events wird diese abgebaut und erst beim Auftreten des konfigurierten Events wieder aufgebaut. Hierfür muss auf der Seite des Clients ein entsprechender Server-Port zur Verfügung stehen.

Nachfolgend werden die Möglichkeiten zur Konfiguration von Events spezifiziert:

Abonnieren einer Event-Benachrichtigung:

Mit Hilfe des Request-Brokers „Fehler und Events“ soll die Überwachung von bis zu 10 Parametern auf das Auftreten eines der folgenden Ereignisse ermöglicht werden:

- Parameterwert ist größer als Vergleichswert
- Parameterwert ist kleiner als Vergleichswert
- Parameterwert ist gleich dem Vergleichswert
- Parameterwert ist ungleich dem Vergleichswert
- Parameterwert weicht vom Vergleichswert weniger als xx Promille ab¹
- Parameterwert weicht vom Vergleichswert mehr oder genau xx Promille ab
- Parameterwert hat sich gegenüber dem Wert zum Konfigurationszeitpunkt geändert

Die 10 Paare aus zu überwachendem Parameter und dem Vergleichswert sind untereinander entweder mit „ODER“ oder mit „UND“ verknüpft.

Die Abtastzeit mit der die Bedingung überprüft werden soll, soll ebenfalls konfigurierbar sein. Bei Auftreten eines Events werden die aktuellen Parameterwerte an den Client geschickt, damit dieser bei ODER-verknüpften Events den auslösenden Parameter feststellen kann.

Kündigen einer Event-Benachrichtigung:

Ein abonnierter Event läuft solange bis er gekündigt wird. Mit Hilfe der „Kündigungs-Nachricht“ kann eine Event Abonnement wieder aufgelöst werden.

3.2 Funktionsweise

Wird noch weiter spezifiziert....

¹ Dies ist besonders für float-Werte wichtig, da Rundungsfehler durch Konvertierungen entstehen können

3.3 Schnittstellen-Spezifikation Request-Broker „Fehler und Events“ – Client

Client → Request-Broker: Events abonnieren:

Die nachfolgende Tabelle 2-1 spezifiziert das eventbasierte Senden von Parametern.

Datum	Stringlänge	Gültigkeit	Beschreibung
BEOT			Beginn des Telegramms
Kennung	20	Zeichen	Kennung des Servers/ Brokers (für Logging-Zwecke) Stringlänge: max_serveerkennung_length
Zeitstempel	30	Zeichen	Zeit zu der das Telegramm abgeschickt wurde Stringlänge: max_zeitstempel_length
Msg-Typ	Keine Begrenzung (k.B.)	$0 < x \leq 3$	Gibt die Art der Message an: 30: Event abonnieren 31: Abonnement kündigen 32: Event eingetreten Gültigkeit: max_msg_typ
Priorität	k.B.	$0 < x \leq 10$	Priorität mit der die Anfrage behandelt werden soll Gültigkeit: max_prio
Verknüpfung	k.B.	$1 \leq x \leq 2$	1: ODER-Verknüpfung zwischen den Parametern 2: UND-Verknüpfung zwischen den Parametern
Polling Intervall	k.B.	$X \geq 0$	Zeitintervall in dem die Daten vom Regier gepollt werden: 0: so schnell wie möglich (x): Intervall in Millisekunden
Anzahl P-IDs	k.B.	$0 < x \leq 20$	Anzahl der zu überwachenden Parameter - Gültigkeit max_pid_anzahl
P-ID Nr. 1	k.B.	$x > 0$	Parameter-ID des ersten zu vergleichenden Parameters
P-Index Nr.1	k.B.	$x \geq 0$	Parameter-Index zum Parameter Nr.1
Event-Typ Nr. 1			1: Parameterwert ist größer als Vergleichswert 2: Parameterwert ist kleiner als Vergleichswert 3: Parameterwert ist gleich dem Vergleichswert 4: Parameterwert ist ungleich dem Vergleichswert 5: Parameterwert weicht vom Vergleichswert weniger als xx Promille ab 6: Parameterwert weicht vom Vergleichswert mehr oder genau xx Promille ab 7: Parameterwert hat sich gegenüber dem Wert zum Konfigurationszeitpunkt geändert
Toleranz Nr. 1			Abweichung in Promille nach oben oder unten
Vergleichswert Nr. 1	30 Zeichen		Vergleichswert mit dem der aktuelle Wert verglichen wird
....			
P-ID Nr. n	k.B.	$x > 0$	Parameter-ID des ersten geforderten Parameters
P-Index Nr.n	k.B.	$x \geq 0$	Parameter-Index zum Parameter Nr. n
Event-Typ Nr. n	k.B.	$0 < x \leq 3$	1: Parameterwert ist größer als Vergleichswert 2: Parameterwert ist kleiner als Vergleichswert 3: Parameterwert ist gleich dem Vergleichswert 4: Parameterwert ist ungleich dem Vergleichswert 5: Parameterwert weicht vom Vergleichswert weniger als xx Promille ab 6: Parameterwert weicht vom Vergleichswert mehr oder genau xx Promille ab 7: Parameterwert hat sich gegenüber dem Wert zum Konfigurationszeitpunkt geändert
Toleranz Nr. n			Abweichung in Promille nach oben oder unten
Vergleichswert Nr. n			Vergleichswert mit dem der aktuelle Wert verglichen wird
EOT			Ende des Telegramms

Tabelle 3-1: Kommunikationsprotokoll zum eventbasierten Lesen von Daten

Die nachfolgende Tabelle 2-1 spezifiziert das Telegramm für das Kündigen der Ereignisbenachrichtigung.

Datum	Stringlänge	Gültigkeit	Beschreibung
BEOT			Beginn des Telegramms
Kennung	20	Zeichen	Kennung des Servers/Brokers (für Logging-Zwecke) Stringlänge: max_serverkennung_length
Zeitstempel	30	Zeichen	Zeit zu der das Telegramm abgeschickt wurde Stringlänge: max_zeitstempel_length
Msg-Typ	Keine Begrenzung (k.B.)	$0 < x \leq 3$	Gibt die Art der Message an: 30: Event konfigurieren 31: Abonnement kündigen 32: Event eingetreten Gültigkeit: max_msg_typ
EOT			Ende des Telegramms

Tabelle 3-2: Kommunikationsprotokoll zum Kündigen der Ereignisbenachrichtigung

Die nachfolgende Tabelle 3-3 spezifiziert das Telegramm zur Benachrichtigung nach dem Eintreten des konfigurierten Event.

Datum	Stringlänge	Gültigkeit	Beschreibung
BEOT			Beginn des Telegramms
Kennung	20	Zeichen	Kennung des Servers/Brokers (für Logging-Zwecke) Stringlänge: max_serverkennung_length
Zeitstempel	30	Zeichen	Zeit zu der das Telegramm abgeschickt wurde Stringlänge: max_zeitstempel_length
Msg-Typ	Keine Begrenzung (k.B.)	$0 < x \leq 3$	Gibt die Art der Message an: 32: Event abonnieren Gültigkeit: max_msg_typ
Anzahl P-IDs	k.B.	$0 < x \leq 20$	Anzahl der zu überwachenden Parameter Gültigkeit: max_pid_anzahl
P-ID Nr. 1	k.B.	$x > 0$	Parameter-ID des ersten zu vergleichenden Parameters
P-Index Nr.1	k.B.	$x \geq 0$	Parameter-Index zum Parameter Nr.1
P-Wert Nr. 1			Parameter-Wert zu PID Nr. 1 beim Eintreten des Events
....			
P-ID Nr. n	k.B.	$x > 0$	Parameter-ID des ersten geforderten Parameters
P-Index Nr. n	k.B.	$x \geq 0$	Parameter-Index zum Parameter Nr. n
P-Wert Nr. n			Parameter-Wert zu PID Nr. n beim Eintreten des Events
EOT			Ende des Telegramms

Tabelle 3-3: Telegramm zur Benachrichtigung bei Eintreffen des konfigurierten Ereignisses

3.4 Schnittstellen-Spezifikation Request-Broker „Fehler und Events“ – Request-Broker „Parameter“

Der Request-Broker „Fehler und Events“ verwendet für die Kommunikation mit dem Regler das in einem gesonderten Dokument spezifizierte BASCOM-Protokoll.

4 Spezifikation des Request-Brokers „Parameter-Bedarfsdaten“

4.1 Spezifizierte Funktionalitäten

Noch zu spezifizieren.

4.2 Funktionsweise

Noch zu spezifizieren.

4.3 Schnittstelle Request-Broker Parameter-Bedarfsdaten – Client

Noch zu spezifizieren.

4.4 Schnittstelle Request-Broker Parameter-Bedarfsdaten – G-Drive

Der Request-Broker „Fehler und Events“ verwendet für die Kommunikation mit dem Regler das in einem gesonderten Dokument spezifizierte BASCOM-Protokoll.

5 Spezifikation des Request-Brokers „Zyklische Sollwerte“

5.1 Spezifizierte Funktionalitäten

Noch zu spezifizieren.

5.2 Funktionsweise

Noch zu spezifizieren.

5.3 Schnittstelle Request-Broker Zyklische Sollwerte – Client

Noch zu spezifizieren.

5.4 Schnittstelle Request-Broker Zyklische Sollwerte – G-Drive

Der Request-Broker „Fehler und Events“ verwendet für die Kommunikation mit dem Regler das in einem gesonderten Dokument spezifizierte BASCOM-Protokoll.

6 Spezifikation des Request Brokers „Software-download“

6.1 Spezifizierte Funktionalitäten

Der Request-Broker „Install/ Setup“ soll das Laden der Firmware auf die Reglerhardware, sowie das Sichern der Firmware von der Reglerhardware ermöglichen.

6.2 Funktionsweise

Noch zu spezifizieren.

6.3 Schnittstelle Request-Broker Softwaredownload – Client

Noch zu spezifizieren.

6.4 Schnittstelle Request-Broker Softwaredownload – G-Drive

Der Request-Broker „Fehler und Events“ verwendet für die Kommunikation mit dem Regler das in einem gesonderten Dokument spezifizierte BASCOM-Protokoll.

7 Spezifikation des Connection-Managers

7.1 Spezifizierte Funktionalitäten

Noch zu spezifizieren.

7.2 Funktionsweise

Noch zu spezifizieren.

8 Weitere Dienste auf dem Kommunikations-PC

8.1 Spezifikation des Logging Servers

8.1.1.1 Funktionsweise

Alle Meldungen, die in den Request-Broker-Prozessen erzeugt werden, werden formatiert und an die Konsole, in ein Logfile oder eine Nachrichtenwarteschlange des Log-Servers geschrieben. Von diesem Log-Server können die Nachrichten dann zu beliebigen Servern / Rechnern verschickt werden.

Um die spätere automatische Auswertung der Logfiles zu ermöglichen, werden verschiedene Nachrichtentypen definiert, die die Interpretation der Nachrichten erleichtern. (z.B. Debug, Daten, Error ...)

Weiteres wird später spezifiziert.

8.2 Spezifikation der Skriptunterstützung

Funktionsweise

Mit Hilfe der Skriptunterstützung auf dem Kommunikations-PC soll eine flexible und frei programmierbare Schnittstellen geschaffen werden, mit der auch zukünftige Anforderungen an Überwachung und Diagnose abgedeckt werden sollen. Mit Hilfe eines Perl-Interpreters können beliebige Skripte auf dem Kommunikations-PC ablaufen, die auf die Funktionalitäten der Request-Broker „Parameter“ und „Fehler und Events“ zugreifen. So können komplexere Überwachungsfunktionen lokal auf dem Kommunikations-PC ausgeführt werden ohne das eine Belastung des Netzwerks durch Übertragung von Daten stattfindet. Die Skripte werden mit Hilfe des ftp-Servers auf dem Kommunikations-PC übertragen oder sind dort schon als Teil der Software vorhanden.

Für den Start und die Datenübertragung zur der das Skript aufrufenden Instanz ist folgendes Protokoll vorgesehen:

Start des Skripts:

Fehlermeldung:

Ergebnisdaten:

8.3 Zeitsynchronisation, FTP-Server, Telnet

Für die Synchronisierung der Systemzeiten sollen alle Kommunikations-PCs ihre Systemzeit regelmäßig über einen auf dem BAUDIS Server laufenden Zeitserver synchronisieren. Der FTP-Server und der Telnet-Zugang dienen für das Software-Update des Kommunikations-PCs und zur Wartung.

Vielachsenanwendungen mit synchronisierter Vernetzung und asynchroner Kopplung an die Leitebene

Ing.(grad) Harold Meis, Dipl.-Ing. Thomas Tschafary
Baumüller Anlagen-Systemtechnik GMBH & CO., Nürnberg,

Kurzfassung

Durch die Nutzung der Einzelantriebstechnik ist die Anzahl der synchron zu betreibenden Antriebe stetig gewachsen. Gleichzeitig erfordert die Notwendigkeit einer effizienten Diagnose einen hohen Datendurchsatz durch alle Ebenen der Automatisierungspyramide. Ein Zusammenwirken der beiden Kommunikationstechnologien SERBAS und Ethernet ermöglicht die Realisierung der Forderung nach hoher Synchronität und einer asynchronen Kopplung an die Steuerungs- und Leitebene. Durch entsprechende Strukturen können so bis zu 10416 Antriebe synchron betrieben werden, wobei die Sollwertvorgabe asynchron von der Leitebene erfolgt. Zur Überwachung dieser großen Zahl von Antrieben sind neue Diagnosekonzepte und Strukturen unabdingbar. Mit BAUDIS NET gibt es einen neuen Lösungsansatz.

1 Vernetzte Antriebe – Vielachsenanwendungen

Der seit einigen Jahren ungebrochene Trend mechanisch gekoppelte Bewegungsabläufe an Fertigungsmaschinen durch einzeln angetriebene Aggregate zu ersetzen, führt zu einer ständig steigenden Zahl von synchronisiert zu betreibenden Antrieben. Die in der Vergangenheit übliche mechanische Leitachse (Königswelle) wird dabei durch eine virtuelle Leitachse ersetzt.

1.1 Anforderungen an die Einzelantriebstechnik

Die Bewegungsabläufe, die es mit der Einzelantriebstechnik nachzubilden gilt, reichen dabei von direktem Leit- Folgebetrieb bis zu komplexen Kurvenverläufen, die von einer realen Leitachse abgeleitet werden müssen. Die Anforderungen an die dynamische und statische Genauigkeit sind dabei beträchtlich. Es gilt die mechanischen Steifigkeiten der mechanischen Leitachse zu erreichen bzw. zu überbieten.

Doch nicht nur die regelungstechnischen Anforderungen sind für den Erfolg solcher Systeme relevant, auch Synchronität und die Konfigurierbarkeit der einzeln angetriebenen Aggregate ist für den Einsatz an Produktionsmaschinen entscheidend. Die Anzahl der synchronisiert zu betreibenden Antriebe kann dabei durchaus mehrere hundert betragen. Typische Beispiele sind in der Druck- und Textilmaschinenbranche zu finden.

Diese große Zahl an Antrieben erfordert naturgemäß ein hohes Maß an Zuverlässigkeit. Auch darf der Aus-

fall eines einzelnen Antriebs oder einer Gruppe von Antrieben nicht zum Ausfall des Gesamtsystems führen.

Kommt es dennoch zu Störungen, dann ist ein System zur schnellen und sicheren Fehleranalyse und Beseitigung unerlässlich. Zur Vermeidung von Störungen sind Methoden zur 'vorbeugenden Wartung' erforderlich. Weiter müssen Diagnose und Servicemaßnahmen von jedem Ort der Welt aus durchführbar sein.

Zusammenfassend sind zwei Anforderungen erkennbar, die entscheidenden Einfluss auf die Kommunikationsstruktur einer Vielachsenanwendung haben: Einerseits muss eine synchronisierte Kommunikation zwischen den Antriebsreglern erfolgen, andererseits ist eine asynchrone Kommunikation mit der Steuerungs- und Leitebene erforderlich.

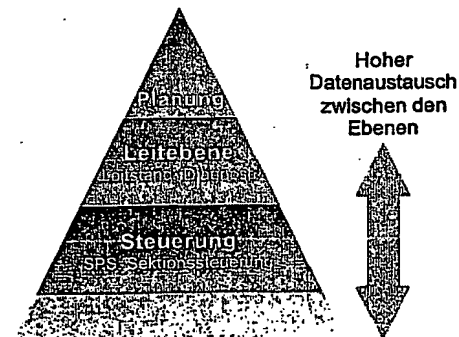


Bild 1 Automatisierungspyramide einer Vielachsenanwendung

Bild 1 zeigt die Automatisierungspyramide einer Vielachsenanwendung. Die Antriebskomponenten in der

Feldebene werden von den Steuerungskomponenten der Steuerungsebene gesteuert. Auf der darüber liegenden Leitebene werden dem Anlagenbediener aufbereitete Daten und Informationen zur Verfügung gestellt, die zur Bedienung und Überwachung der Anlage verwendet werden. Die Spitze der Pyramide bildet die Planungsebene. Aufgrund der oben genannten Anforderungen einer Vielachsenanwendung ist ein hoher Datenaustausch zwischen den einzelnen Ebenen der Automatisierungspyramide erforderlich. Hierzu kommen in der Baumüller SyncDrive-Technologie zwei sich ergänzende Kommunikationstechnologien zum Einsatz: SERBAS¹ und Ethernet mit TCP/IP und UDP/IP.

2 Synchronisierte Kommunikation mit SERBAS

Um eine Vielzahl von Antrieben (z.B. eine Zeitungsdruckmaschine mit bis zu 400 Antrieben) hochgenau synchronisiert zu betreiben, sind auf diesen Anwendungsfall zugeschnittene Lösungen unerlässlich. Daher kommt für die synchronisierte Kommunikation auf der Feldebene eine stark an SERCOS-Interface angelehnte Kommunikationstechnologie zum Einsatz. In den folgenden Abschnitten wird SERBAS näher erläutert.

2.1 SERBAS Ringstrukturen

2.1.1 Der SERBAS Antriebsring

Der SERBAS-Antriebsring besteht aus einem Kommunikations-Master mit Leitachsfunktionalität (MDS) und aus bis zu 48 Antriebsreglern (MDC). Üblicherweise wird eine mechanisch funktionelle Einheit mit einem Antriebsring realisiert. Der Kommunikations-Master hält dabei die Anbindung zur Steuerungsebene und verteilt die Sollwerte sowie Steuerinformationen an die betroffenen Antriebsregler. Ebenso werden von den Antriebsreglern Istwerte und Statusinformationen für die Steuerungsebene bereitgestellt. Auf diese Art ist ein Baustein entstanden, der einzeln in Betrieb genommen und getestet werden kann. Bild 2 zeigt einen SERBAS-Antriebsring mit einem Kommunikationsmaster und 9 Antriebsreglern. Insgesamt beinhaltet dieser Ring 10 geregelte Antriebseinheiten. Die gestrichelten Verbindungen zwischen den Antriebseinheiten stellen die Redundanzfunktion in der SERBAS-Kommunikation dar. Sie wird in Kapitel 2.3 erklärt.

¹ SERBAS ist eine hochgenaue synchronisierte serielle Schnittstelle in Erweiterung von SERCOS-Interface

2.1.2 Die SERBAS Querkommunikation

Die Verknüpfung der Antriebsringe geschieht durch einen zweiten SERBAS-Ring. Dazu steht im Kommunikations-Master (MDS) eine zweite SERBAS-Schnittstelle zur Verfügung. Mit dieser wird eine Querverbindung zwischen den Kommunikations-Mastern hergestellt. Auf diese Weise können bis zu 32 Teilnehmer, d.h. Kommunikationsmaster, in der Querkommunikation miteinander verbunden werden. Auf dieser Ebene werden die Sollwerte von bis zu 32 Leitachsen mit den zugehörigen Steuer- und Statusinformationen zur Verfügung gestellt. Dabei kann jeder Antrieb einer beliebigen Leitachse zugeordnet werden. Es besteht weiterhin die Möglichkeit die Antriebe in Gruppen zusammen zu fassen. Damit wird eine flexible Anpassung an die geforderten Produktionsbedingungen möglich.

So ist ein neuer Baustein entstanden, der bis zu $32 \times 48 = 1538$ Antriebe synchronisiert betreiben kann.

Bild 3 zeigt die SERBAS-Querkommunikation zusammen mit den Antriebsringen aus Bild 2.

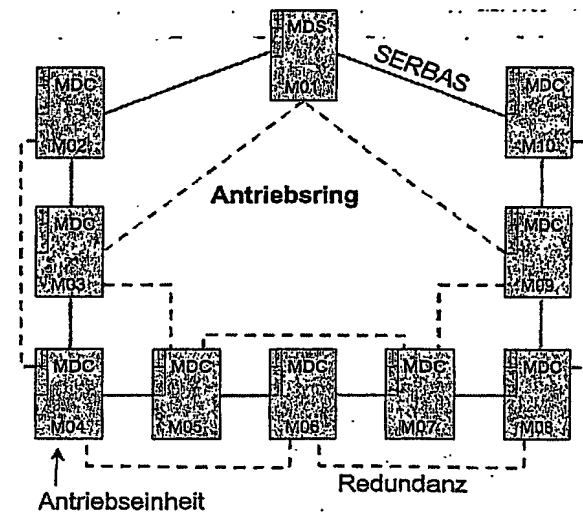


Bild 2 Der SERBAS Antriebsring

2.1.3 Der Multi-Link-Controller

Um die Kommunikationsstruktur der Maschinenstruktur anzupassen und die Anzahl der miteinander synchronisiert kommunizierender Bausteine zu erhöhen, wurde in der Baumüller Sync-Drive Technologie ein weiteres Strukturelement eingeführt: der Multi-Link-Controller (MLC).

Der Multi-Link-Controller wird als einer der 32 möglichen Teilnehmer in den Querkommunikationsring aufgenommen. Das ist zur Zeit bei bis zu 7 Querkommunikationsringen möglich. Der MLC verteilt dabei die für den synchronisierten Betrieb aller angeschlossenen Ringe relevanten Daten. Im wesentlichen

sind das alle Leitachsinformationen der 32 im Gesamtsystem möglichen Leitachsen, so dass jeder der im System vorhandene Antriebe einer dieser Leitachsen zugeordnet werden kann. Bild 4 zeigt die Koppelung von bis zu 7 Antriebssektionen mit Hilfe des Multi-Link-Controllers.

Durch den Einsatz des MLC erhöht sich die Anzahl der synchronisiert zu betreibenden Antriebe auf $48 \times 31 \times 7 = 10416$. Weit wichtiger als diese große Zahl an Antrieben ist die Möglichkeit, das Antriebssystem und seine Kommunikationsstruktur an die Struktur der zu betreibenden Maschine anzupassen und damit auch komplexe Antriebssysteme beherrschbar zu machen.

Einzelne an den MLC gekoppelte Kommunikationsringe können aus dem Gesamtsystem entfernt und wieder hinzugefügt werden ohne den restlichen Teil der Anlage zu beeinflussen. Ein wichtiger Aspekt für Inbetriebnahme und Wartung.

2.2 Sollwertgenerierung und Verteilung

Durch den Einsatz von dezentralen Sollwertgeneratoren ist es nicht erforderlich, zentrale Lagesollwerte zu generieren und im System an alle Antriebe synchronisiert zu verteilen. Stattdessen muss eine in der Leitebene angesiedelte Sollwertquelle nur bei Änderung neue Drehzahl- und Beschleunigungssollwerte bereitstellen. Diese Sollwerte werden dann in einem System aus strukturierten SERBAS-Ringen an die Antriebe verteilt, und, synchronisiert durch den SERBAS-Takt, dezentral zu Lagesollwerten umgerechnet. Die Synchronität der Antriebsregler ist entscheidend für das erreichbare Ergebnis. Ein Beispiel aus der Applikation Zeitungsdruck: Eine zeitliche Verschiebung des Sollwerts um $1\mu\text{s}$ führt bei einer Druckgeschwindigkeit von 35.000 Exemplaren/h zu einem Winkelfehler von 3,5 mGrad. Auf dem Papier kann das einen Versatz von 0.01 mm bewirken. (Umfang der Druckwalze ca. 1.100 mm)

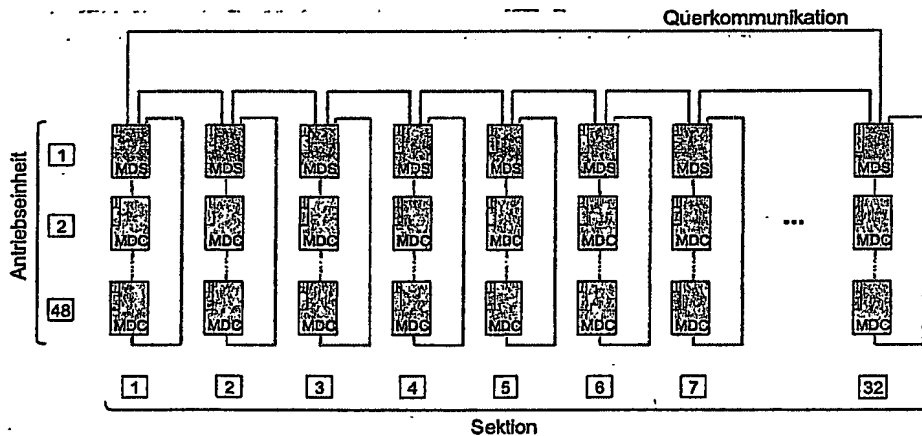


Bild 3 Die SERBAS-Querkommunikation

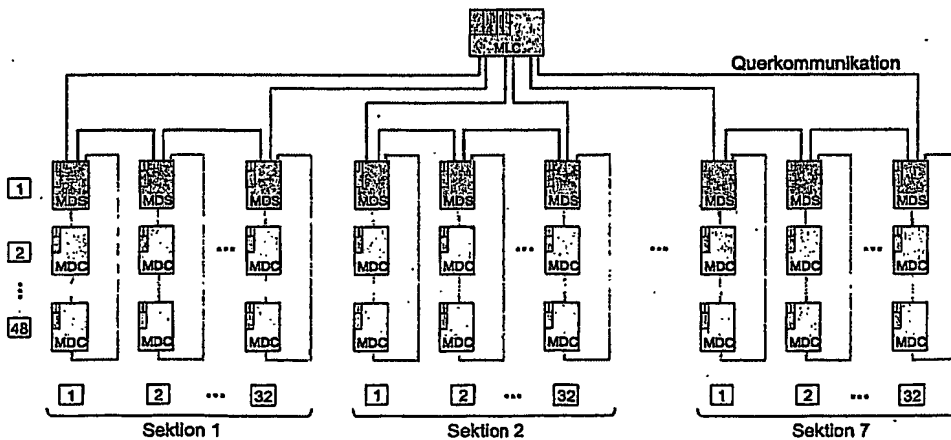


Bild 4 Koppelung von bis zu 7 Antriebssektionen mit Hilfe des Multi-Link-Controllers

2.3 Die Redundanzfunktion in der SERBAS-Kommunikation

Durch die SERBAS-Ringstruktur würde der Ausfall eines Teilnehmers zur Unterbrechung der Kommunikation und damit zum Ausfall des gesamten betroffenen Antriebsrings führen. Der Forderung nach einem robusten System wird durch eine redundant ausgeführte SERBAS-Struktur entsprochen. Dabei wird ein zweiter SERBAS-Ring realisiert, bei dem jeweils der übernächste Teilnehmer miteinander verbunden wird. Durch diese Struktur wird sichergestellt, dass jeder zweite Teilnehmer ausfallen kann, ohne die Kommunikation der restlichen verbliebenen Teilnehmer zu gefährden. Die Redundanzfunktion ist konsequent für alle SERBAS-Ringe realisiert. Die gestrichelten Verbindungen in Bild 2 zeigen die redundante SERBAS-Ringstruktur.

3 Asynchrone Kommunikation mit Ethernet

Die Verbindung zwischen Feldebene und der Steuerungsebene sowie zwischen Feldebene und der Leitebene in der Automatisierungspyramide aus Bild 1 wird durch asynchrone Kommunikation über Ethernet realisiert. Es kommen dabei die Protokolle TCP/IP und UDP/IP zum Einsatz. Im folgenden werden die verschiedenen Anforderungen an die asynchrone Kommunikation erläutert.

3.1 Kommunikation zur Steuerung

Bei der Kommunikation zwischen Feldebene und Steuerungsebene wird zwischen zyklischen Daten und Bedarfsdaten unterschieden. Bild 5 (links) zeigt die Kommunikation mit der Steuerungsebene. Wie Bild 5

(links) zeigt ist es bei Verwendung von Ethernet einfach möglich Sollwerte von mehreren Steuerungen (SPS) zu erhalten.

3.1.1 Zyklische Kommunikation

Zu den zyklischen Daten gehören Soll- Istwerte sowie Steuer- und Statusinformationen. Diese werden mit UDP/IP übertragen. Da bei dem hier genutzten verbindungslosen Protokoll eine sichere Übertragung nicht gewährleistet ist, werden ausschließlich Daten übertragen, deren Verlust den Betrieb der Maschine nicht gefährdet. Durch den im Vergleich zu TCP geringeren Protokoll-Overhead, können kleine Übertragungszyklen gewählt werden. Ein Ausfall von 2 UDP-Telegrammen kann dann toleriert werden ohne den Betrieb der Maschine zu gefährden. Der Kommunikations-Master in der Antriebsebene (MDS) stellt somit die Schnittstelle zwischen der asynchronen Kommunikation zur Steuerungsebene und der synchronisierten Kommunikation in der Antriebsebene dar.

3.1.2 Bedarfsdatenkommunikation

Das verbindungsorientierte Protokoll TCP wird zur Übertragung von Daten zur Konfiguration bzw. zur Parametrierung des Antriebssystems eingesetzt. Um den Preis des erhöhten Overhead (ca. 15 Mal mehr als bei SERBAS) kann mit TCP von einer gesicherten Übertragung ausgegangen werden.

Auch hier findet die Kommunikation zwischen Steuerungsebene und Kommunikations-Master (MDS) in der Antriebsebene statt. Die Verteilung der Informationen in der Antriebsebene erfolgt mit Hilfe des SERBAS-Protokolls.

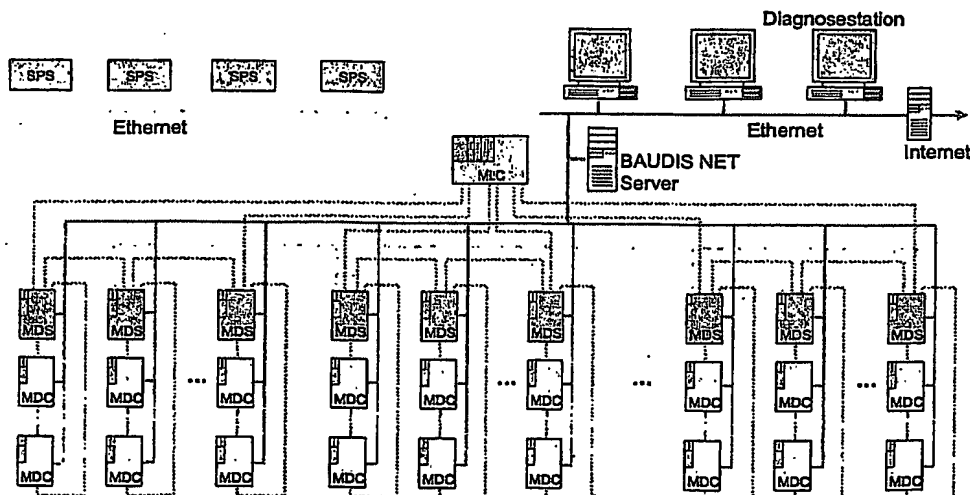


Bild 5 Asynchrone Kommunikation mit der Steuerungsebene (SPS) sowie der Leitebene (Diagnose)

3.2 Kommunikation für die Anlagenüberwachung und -Diagnose

Mit der wachsenden Zahl von Antrieben werden Funktionen zur Inbetriebnahme und Überwachung der Antriebe unentbehrlich. In der Leitebene werden ständig Informationen aus der Antriebsebene benötigt. Dabei handelt es sich im wesentlichen um Systeminformationen wie Status- und Fehlermeldungen, Wartungsinformationen sowie Aufzeichnungen zur Qualitätsüberwachung.

Zur Auswertung der Daten steht in der Leitebene ein Diagnoserechner (BAUDIS NET Application Server) zur Verfügung. Auch hier wird Ethernet als Kommunikationsmedium eingesetzt. Das OSI-Schichtenmodell ermöglicht komplexe Kommunikationsmechanismen zwischen Antrieb- und Leitebene. Da die Diagnose unabhängig von der restlichen Kommunikation erfolgen soll, (schließlich sollen u.a. auch Kommunikationsprobleme in den SERBAS-Ringen erkannt werden) muss jeder Antriebsregler über Ethernet erreichbar sein.

3.2.1 Dezentrale Diagnose

Um die anfallende Datenmenge sinnvoll verarbeiten zu können, ist eine dezentrale Vorverarbeitung erforderlich. Ebenfalls sinnvoll ist es, umfangreiche Diagnosefunktionen möglichst nahe am betroffenen Gerät anzusiedeln. Zu diesem Zweck ist in jedem Antriebsregler ein Kommunikations-PC integriert. Dieser übernimmt neben der Anbindung der Antriebsregler an das Ethernet auch die Event-basierte Kommunikation zum BAUDIS-Server. Bild 6 zeigt die Systemstruktur von BAUDIS NET.

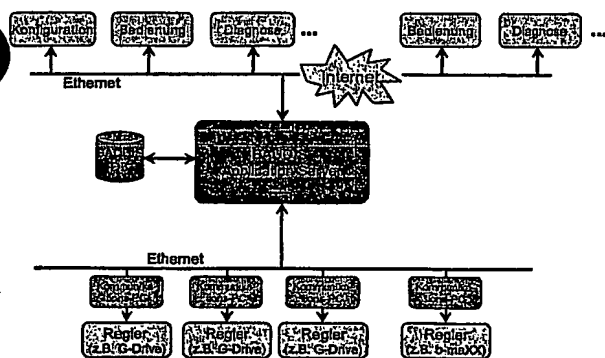


Bild 6 Systemüberblick BAUDIS NET

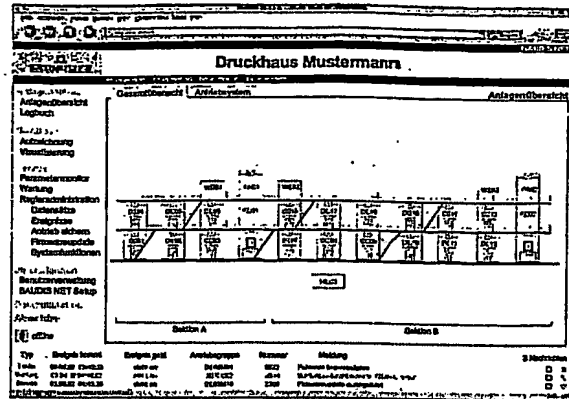


Bild 7 Anlagenorientiertes Anlagenbild in BAUDIS NET (am Beispiel einer Druckmaschine)

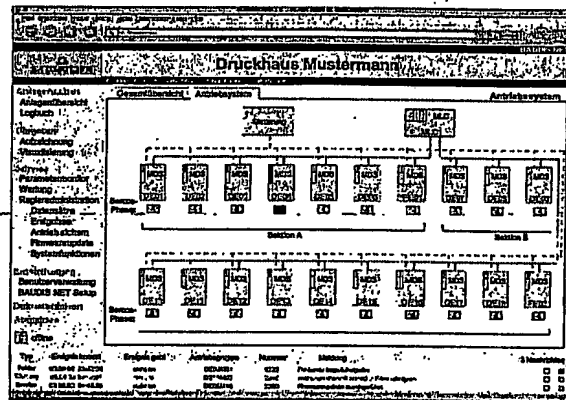


Bild 8 Antriebsystem-orientiertes Anlagenbild in BAUDIS NET (am Beispiel einer Druckmaschine)

3.2.2 Webbasierte Bedienoberflächen

Die Information, die das oben beschriebene Diagnosesystem zur Verfügung stellt, wird an vielen unterschiedlichen Orten innerhalb und außerhalb der zu überwachenden Anlage benötigt. Da ist z.B. der Maschinenleitstand mit dem Bedarf an Daten für den aktuellen Maschinenstatus, oder die Produktionsleitung mit Bedarf an statistischen Daten zur Maschinenverfügbarkeit und Wartungszyklen. Aber auch der Maschinenhersteller bzw. der Antriebsausrüster muss auf jede ausgelieferte Anlage zugreifen können, um im Servicefall schnelle und effiziente Diagnose und Fehlerbeseitigung zu gewährleisten. Dabei ist es völlig unerheblich in welchem Teil der Welt die Anlage installiert ist.

Um all diese Forderungen zu erfüllen, ist der Einsatz von modernen Web-Technologien nahezu unabdingbar. Webbasierte Bedienoberflächen bieten aufgrund ihrer Flexibilität gegenüber konventionellen Oberflächen einige entscheidende Vorteile:

- Die Weboberfläche kann auf beliebigen Client-Rechnern laufen (unabhängig vom Client Betriebssystem)
- Eine Installation der BAUDIS NET Bedienoberfläche auf dem Client-Rechner entfällt
- Weboberflächen sind aufgrund ihrer weiten Verbreitung durch das Internet leicht bedienbar
- Die Bedienoberfläche kann mit vertretbarem Aufwand an Kundenwünsche angepasst werden (z.B. Sprachunterstützung)

Bild 7. und Bild 8 zeigen zwei Weboberflächen, die die benötigte Information für den entsprechenden Benutzerkreis graphisch aufbereiten.

Erst die Bereitstellung der richtigen Informationen (bezogen auf den jeweiligen Nutzer) zum richtigen Zeitpunkt und am richtigen Ort, macht dieses Diagnosesystem zu einem Instrument, dass die Maschinenverfügbarkeit erhöht und auch komplexe Anlagen mit einer Vielzahl von Antrieben beherrschbar macht.

Wie Bild 9 zeigt, sind die an der Anlage aufbereiteten Informationen mit Hilfe von BAUDIS NET sowohl lokal als auch an einem beliebigen Ort verfügbar.

4 Zusammenfassung

Die oben beschriebenen Eigenschaften der Baumüller Sync-Drive Technologie kommen immer dann besonders zur Geltung, wenn es darum geht eine Vielzahl von Antrieben mit sehr hohen Ansprüchen an Dynamik und Synchronität zu betreiben. Für die Kommunikation durch die Ebenen der Automatisierungspyramide hat sich das Ethernet weitgehend als Standard durchgesetzt.

Nur durch die Nutzung der Stärken der unterschiedlichen Kommunikationsmedien ist es möglich, den immer höher werdenden Ansprüchen in den jeweiligen Automatisierungsschichten gerecht zu werden. Wie gezeigt wurde, ist die Transparenz von der Leitebene bis in die Antriebsebene durch BAUDIS NET dabei vollständig gewährleistet.

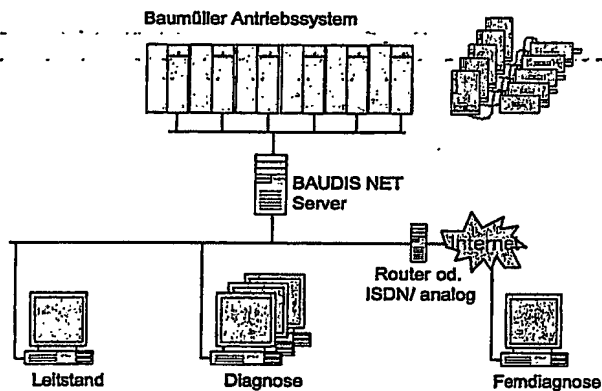


Bild 9 Lokaler und weltweiter Zugriff auf die Diagnoseinformationen durch BAUDIS NET

**Besprechung am 06.03.2002 bei Baumüller
Anlagen-Systemtechnik GmbH & Co. zwischen
Herrn PA G. Götz, Herrn Meis und Herrn
Tschaftary**

M: Ich kann Ihnen das auch mitgeben, denke ich. Dann haben Sie da auch noch eine Grundlage. Ja, worum geht's? Es geht um ein Diagnosesystem und wir merken einfach, dass die Anforderungen immer höher werden. Ich habe hier mal so ein paar Anforderungen zusammengestellt. Wir rechnen damit, dass in den nächsten Jahren die Anzahl von Antrieben, die wir diagnostizieren sollen, wesentlich steigt in Richtung von 400, 500 Antrieben.

G: Bisheriger Stand?

M: 250.

G: 250 Antriebe.

M: 250, 280. Das ist, glaube ich, im Moment ein Maximum. Und das erfordert natürlich zum einen weit mächtigere Funktionen, das erfordert, dass wir auf dem Bereich der Inbetriebnahme mehr tun und so versuchen, eine Anlage schneller in Betrieb zu nehmen und dass wir natürlich auch unsere Systemstruktur, die wir jetzt haben beim BAUDIS irgendwo anpassen müssen.

G: Also, Sie verändern jetzt die Systemstruktur auch?

M: Ja. Wir machen zwar grundsätzlich dasselbe. Wir haben viele alte Funktionen, eigentlich. Alle alten Funktionen, die wir im jetzigen BAUDIS haben, die werden wir auch im neuen BAUDIS haben. Für den User schaut das genauso aus, aber wir realisieren letzten Endes ganz anders.

G: Jetzt habe ich mal eine kurze Frage. Wir haben ja hier so ein grobes Blockschaltbild damals gehabt von dem BAUDIS. Sagt das Ihnen jetzt hier was?

M: Das habe ich jetzt selber noch nicht angesehen.

G: Na, dann lassen wir das jetzt weg. Das ist jetzt nicht ergiebig. Mich hätte jetzt nur interessiert, ob Sie dieses im wesentlichen beibehalten.

M: Das zeige ich Ihnen gleich, wie das dann aussieht. Dann können wir das gegeneinander stellen. Also, die Anforderungen steigen hier, wir haben auch noch die Anforderung hinzubekommen, dass unser Kunde diese Daten des Diagnosesystems auch noch mehr betrachten möchte und in einer Form zur Verfügung gestellt haben möchte, wie es für ihn notwendig ist. Wir haben den weltweiten Zugriff des alten BAUDIS eigentlich auch schon. Und wir wollen, und das ist unser Ziel, letzten Endes dahin, dass wir neue Methoden und Funktionen in dieses BAUDIS hineinbringen, die uns eben helfen, die Antriebe besser zu überwachen. Fehler schneller zu finden, solche Dinge. Das ist also dieser ganze Strauß von Anforderungen und ich glaube so richtig interessant wird es eigentlich dann hier, weil hier sieht man jetzt mal die Struktur von diesem neuen System. Was natürlich alt ist, ist die Antriebsebene, d. h. also hier sind die ganzen Antriebe mit ihren Reglern symbolisiert.

G: Also „G“ heißt Giga-Drive.

M: Ja.

T: ...

G: Ach, so der ist jetzt Giga, vorher war es Mega-Drive.

T: Richtig, ja.

M: Das ist jetzt ein Arbeitstitel.

G: Was kommt nach Giga? Wissen Sie noch nicht.

T: Tera, gibt es auch noch, oder?

G: Tera-Drive.

T: Das Giga leitet sich einfach davon her, dass der einen Prozessor hat, der Giga-Flops kann und von daher war das naheliegend, den Titel zu wählen.

G: „Flops“ ist Abkürzung für ...

T: ...

M: O.k. Da gibt es also diese Antriebsebene, es läßt sich aber auch denken, dass hier andere Reglertypen von BAUMÜLLER drin sind, z. B. dieser b maXX, der neu entwickelt wird in der Elektronik, das ist auch eine Neuerung gegenüber dem alten BAUDIS. Und um diese ganze Diagnose zu bewerkstelligen, haben wir für jeden G-Drive im

Moment einen Kommunikations-PC geplant. Also, PC ist da zuviel gesagt. Also, das ist einfach nur so ein kleines Platinenmodul, wo ein Prozessor drauf sitzt und das wird auf die Hardware des Reglers draufgesteckt. Ist aber ein vollwertiges Linux, so ein kleines Linux. Und nach außen schaut es so aus, als wäre es ein ganzer PC.

G: Also, mit Gehäuse und so darf man sich das nicht vorstellen, mit eigenem Bildschirm, sondern es ist bloß eine Elektronikarte mit Prozessor drauf.

M: Richtig.

G: Linux-Betriebssystem.

M: Richtig. Also, nur so ein kleiner Typ mit einer Ethernet-Schnittstelle und einer Schnittstelle zur anderen Seite. Und was wiederum gleich geblieben ist, wir haben hier so einen zentralen Rechner, den nennen wir jetzt BAUDIS Application Server, hier sind alle Funktionen, die wir so für die Diagnose und Inbetriebnahme brauchen, zu finden. Der ist angekoppelt an der Datenbank, die natürlich auch auf diesem Rechner läuft. Also, da ändert sich jetzt so von außen her nichts, bloß die Architektur, das was da drin steckt, das wird quasi völlig anders.

G: Also, in diesem großen blauen Kästchen jetzt auf der Seite 3 hier da ändern Sie innen drin die Struktur. Von diesem sog. BAUDIS-Application-Server.

M: Ja. Vorher war das der BAUDIS-Server und jetzt nennen wir den Application-Server, um das ein bißchen zu unterscheiden und weil das der landläufige Begriff dafür ist. O. K. Und jetzt haben wir eine Reihe von Anwendungen hier auf dieser Ebene. Also, es sind verschiedene Anwendungen denkbar. Anwendungen, die so ein Antriebssystem konfigurieren, die so ein Antriebssystem bedienen oder die diese Diagnose durchführen. Und wir können sowohl wenn wir vor der Anlage stehen als auch wenn wir über das Internet oder Telefon uns mit dieser Anlage verbinden. Und das neue daran ist, das ist ein wesentliches Merkmal, dass wir quasi die Oberbleche, also das, was der Benutzer sieht, und wo er seine Eingaben macht, trennen von der Funktion, da wo es implementiert ist, vom Server quasi.

G: Und worin sind dann diese Oberflächenfunktionen implementiert? In welcher Hardware?

M: Das läuft dann quasi auf einem separaten PC, also z. B. auf dem Leitstandsrechner von dieser Druckmaschine oder auf irgend einem

anderen PC in einem Webbrowser. Also, das was Sie kennen als Internet Explorer oder Netscape Navigator, da kann man diese Oberfläche aufrufen und kann alle Funktionen bedienen, die hier auf dem Application-Server ausgeführt werden.

G: Und da ist zwischengeschaltet ein Bus oder so was? An Ethernet ist ja im Prinzip ein Bussystem. Ist das richtig, ja?

M: Das ist ein lokales Netz, wie wir hier auch haben.

G: Ein lokales Netz.

M: ... vernetzt sind. Lokales Netz mit Anschluß an das Internet bzw. über Modem kann ich mich dann hier in dieses Netz einwählen und komme dann an diesen Rechner.

G: Kann man das so ausdrücken: Sie lagern eigene Netzwerkknoten aus, um diese Bedienoberfläche zu realisieren? Das heißt, für die Bedienoberfläche werden innerhalb eines Netzwerks eigene Netzwerkknoten ausgebildet.

M: Hm, ja, das ist etwas ungewöhnlich formuliert, aber ...

G: Kann man mal so stehenlassen, vielleicht. Ich kenne es ja jetzt nicht.

M: Also, es ist genau dieselbe Struktur wie das, was Sie haben, wenn Sie eine Internetseite aufrufen. Wenn Sie jetzt auf „www.kugel.de“ oder so etwas ...

G: Auf „www.kugel?“

M: „kugel.de“ oder irgend etwas, was Ihnen einfällt. „telekom.de“ oder so etwas. Dann rufen Sie ja eine Seite auf und der Inhalt dieser Seite wird abgerufen von so einem Server.

G: Ah, ja. Und Sie übernehmen jetzt quasi das Internetmodell.

M: Richtig, ja. So könnte man das sagen.

G: Und dann ist der BAUDIS-Application-Server der zentrale Server und ich rufe quasi dessen Internetadresse auf oder dessen Netzwerkadresse auf.

M: Ja, dessen Netzwerkadresse und dann ist das quasi genauso wie wenn ich so eine Seite von der Telekom aufrufe. Grob vereinfacht, könnte man das so sagen.

G: Und das sind das hier die Clients, oder was? So nennt man das, glaube ich, bei Ihnen?

M: Richtig, das sind dann die Clients und das ist der Server. Und wir müssen das nicht unbedingt über das Internet machen bzw. über eine Modemverbindung, wir können das natürlich auch machen, auf allen Rechnern, die jetzt hier im Netz angeschlossen sind.

G: Und bisher waren diese Funktionen Diagnose, Bedienung, Konfiguration, die hier über das Ethernet an den Server angeschlossen sind im Server selber integriert, sind dort abgelaufen. Und ich habe keine Ethernetkommunikation dazwischen gehabt.

T: Nicht zur Fernbedienung; naja, jein ...

M: Doch, doch das hatten wir eigentlich schon. Doch wir hatten das eigentlich schon. Im Normalfall ist das so, dass derjenige, der das BAUDIS bedient, an diesen Rechner geht. Wir haben aber zusätzlich die Möglichkeit der Ferndiagnose. Deswegen haben wir dieses System ja entwickelt, nämlich dass wir uns z. B. vom einem entfernten Rechner, z. B. von dem hier im Büro über eine Modemverbindung an diesen BAUDIS-Rechner anhängen können und dann auch dasselbe tun.

G: Das war mal hier früher so ähnlich, aber ich habe schon fast das Gefühl, ob das überhaupt so richtig gestimmt hat mit dem realen BAUDIS. Das BAUDIS-Patent, das ist auch die Frage. Das war wahrscheinlich ganz in den Anfangsgründen und so und dann hat es sich jetzt schon sehr stark von dem Patent wegentwickelt. Gut. Dann bleiben wir bei Ihnen, ich glaube, das Patent hier ist gar nicht so hilfreich.

T: Unterschiedlich ... ist das hier schon ein Unterschied zu dem Application-Server und zu dem hier oben, glaube ich, dass der entfernte PC sogar jetzt genutzt wird, halt auch die BAUDIS-Applikation haben mußte, die hängt auf diesem entfernten System. Und das ist hier nicht mehr notwendig ... Und das ist der entscheidende Unterschied.

G: Ach, so. Die Applikation läuft jetzt nicht mehr auf diesem Netzwerkknoten für den Bediener selbst, sondern der Netzwerkknoten ruft sich, immuliert (?) die Applikation, kann man das sagen?

T: Nö, nö. Der kriegt einfach Daten. ...

G: Also, genau so, wie wenn der Internetnutzer eine Website aufruft.

M: Richtig.

G: Das Model übertragen Sie jetzt hierauf.

M: Genau, ja. Das bietet uns eben Flexibilität, das bietet uns, dass wir dem Kunden die Informationen anzeigen können, wo er es möchte. Ohne dass wir auf dem PC, wo er es möchte, irgend etwas installieren müssen oder irgendwie den verändern müssen, oder so was. Sondern wir sagen eben: Du brauchst bloß so einen Internetbrowser haben, dann meldest Du Dich hier an und dann kriegst Du die ganzen Informationen. Das ist quasi für den Nutzer so das Entscheidende. Also, in der Struktur neu ist dieser Kommunikations-PC, neu ist die Trennung von Oberfläche und Server.

G: Die Oberfläche die bilden jetzt einzelne Netzwerkknoten, Clients, Client-Knoten, die dann über das Ethernet mit dem Server verbunden sind.

M: Ja, denke ich, das kann man so sagen.

G: Dieses Bild steht dann für die Patentanmeldung, ist das vielleicht ganz gut, ganz brauchbar.

M: Wenn wir dann so weit kommen, dann können wir das natürlich verwenden. Ja und jetzt, ich weiß nicht, inwieweit das jetzt Sinn macht, aber noch einmal ein bißchen hineinzuschauen in diese einzelnen Komponenten hier, was wir hier neu haben.

G: Je mehr Merkmale wir haben, um so mehr Sicherheit haben wir für eine Patentanmeldung und bessere Chancen.

M: Vielleicht sollten wir das einfach mal kurz machen. Wir schauen jetzt mal in diesen Kommunikations-PC hinein, was da drin abläuft. Dieser Kommunikations-PC hat quasi die Aufgabe, alles was an den Regler, der hier oben jetzt nicht gezeichnet ist, aber hier oben würde man ihn dann sehen, alles was an die Regler an Informationen geschickt werden muß oder vom Regler an Information geholt werden muß, abzuwickeln. Also, Kommunikation erledigt er. Und da gibt es verschiedene Dinge, die übertragen werden müssen. Fangen wir mal mit diesen beiden Kästen an hier, das ganze ist also eine modulare Softwarestruktur, also alles was man hier sieht symbolisiert ein Stück Software, was auch im Windows-Betriebssystem läuft. Und hier gibt es also zwei Softwaremodule, die erledigen die Kommunikation mit der Steuerung, also irgendwo muß ja diese

Maschine auch gesteuert werden und dazu müssen spezielle Sollwerte und andere Parameter übermittelt werden und das wird von diesen beiden Modulen erledigt. Die nennen wir Request-Broker. Broker deswegen, weil sie Informationen austauschen, von hier von diesen Clients, also das wäre in dem Fall entweder dieser BAUDIS-Application-Server, der ist jetzt gegenüber dem natürlich als Client aufzufassen, weil er ja von dem was will und in diesem Fall hier wäre das die Steuerung. Die Steuerung möchte nämlich auch unserem Regler sagen, bitte ...

G: Die Steuerung haben Sie hier, glaube ich, auch aufgezeichnet.

M: Habe ich hier aufgezeichnet, aber in einem anderen Zusammenhang.

G: Hat die dann eine direkte Verbindung zu so einem Kommunikations...?

M: Nein.

G: Hat sie nicht. Das geht über Server.

M: Das ist hier ein anderer Zusammenhang. Damit ist jetzt nicht die Steuerung gemeint, von der ich jetzt gerade ...

G: Ach, so! Diese Steuerung auf Seite 3 hat nichts mit der Steuerung von der Seite 17 zu tun.

...

M: Man könnte das auch hier im ersten weglassen und müßte dann näher, warum es dann hier ist.

T: Und die hat dann durchaus Zugang direkt zu dem PC.

M: Ja, ja. Genau. Also die Steuerung wäre eine von diesen Clients, die was von diesem Kommunikations-PC wollen und die schickt ihm über diese beiden Module Sollwerte. „Fahr mal so schnell“ oder „mach mal dieses, mal jenes“. Und jetzt haben wir hier diesen roten Broker, den nennen wir Parameterschnittstelle. Dazu muß man sagen, dass unser G-Drive eigentlich nichts weiter versteht als Parameter. Der hat einen Satz von ungefähr 1500-2000 Parametern und über diese Parameter kann man alles einstellen. Alles, was man mit diesem tun kann, muß man über Parameter einstellen. Und dieses Softwaremodul ist über eine Schnittstelle hier mit dem G-Drive verbunden und kann Parameter lesen und schreiben. Wesentlich mehr kann es nicht. Also wenn irgendeiner dieser Clients hier irgendeinen Parameter lesen möchte oder eine Gruppe von

Parametern, dann bedient er sich dieser Parameterschnittstelle hier. Er gibt quasi den Auftrag und sagt „Ich möchte jetzt den Parameter 4711 und 4712 möchte ich lesen“, dann wird hier ein entsprechender Softwareprozeß gestartet, der fordert jetzt diese Parameter von dem G-Drive an, der G-Drive schickt sie ihm und dieser Request-Broker gibt sie dann weiter an den, der das aufgerufen hat. So hat man sich das vorzustellen. Und man kann mit diesem Request-Broker auch noch andere Dinge machen. Man kann dann sagen, ich möchte den Parameter 4711 im Sekundenabstand geschickt bekommen.

G: Ist das was neues, dieser Kommunikations-PC, diese Karte?

M: Ja.

G: Dann müßte man da wahrscheinlich einen eigenen Patentanspruch oder so was da draufsetzen, dass man den für sich genommen allein schon geschützt hat. Natürlich dann auch im Hinblick auf das Gesamtsystem aber auch schon ohne das Gesamtsystem, eventuell. Gut, das denke ich mir jetzt nur so, wie man das eventuell konzipieren könnte für eine Patentanmeldung.

M: Gut. Dann gibt es noch ein paar andere Module hier. Ich weiß nicht, ob es Sinn macht, die alle zu erläutern. Dieser hier kümmert sich z. B. um Fehler. Jetzt ist es ja so, die Hauptaufgabe von dem Diagnosesystem ist, schnell rauszubekommen, wann wo der Fehler auftritt an so einer Anlage und dieser Request-Broker ist dafür zuständig. Der kuckt also immer bestimmte Parameter, die man eben konfigurieren kann, nach und schaut, ob da ein Fehler aufgetreten ist. Und wenn ein Fehler aufgetreten ist, dann meldet er denjenigen, die sich dafür interessieren, dass so ein Fehler oder ein Ereignis aufgetreten ist. Und das neue oder das schöne an diesem Broker ist, dass ich den halt frei konfigurieren kann. Ich kann mir beliebige Fehler oder beliebige Ereignisse konfigurieren. Ich kann sagen, also wenn der Parameter X auf 10 steht, dann möchte ich das wissen. Und dann sagt er mir das.

G: Also, wenn wir das Bildchen bringen, müssen wir schon alle Kästen erklären.

M: Natürlich. Ich versuche jetzt bloß, das Notwendigste zu erklären, damit wir da nicht durcheinander kommen. Das ist also jetzt eine modulare Softwarestruktur. Man sieht z. B. hier noch, dass dieser Broker von der Steuerung nicht direkt auf den G-Drive zugreift, da fehlt ja die Linie, sondern der bedient sich im Prinzip dieser Parameterschnittstelle, um darauf zuzugreifen.

G: Ach, ja. Das ist ein bißchen codiert gezeichnet.

M: Also, hier diese rote Linie da, die wird quasi zu jedem ... Also jeder kann diese Parameterschnittstelle ... weil letzten Endes möchte die Steuerung auch bloß Parameter lesen und schreiben.

G: Braucht man vielleicht auch weniger Verkabelung.

M: Softwaremäßige Verkabelung braucht man weniger, genau. Und der Vorteil ist halt, dass ich hier diesen Block nur ein einziges mal schreiben muß. So müßte ich jetzt das, was hier drin steckt, hier schreiben und hier schreiben und hier schreiben und hier schreiben und dann sage ich einfach, O. K. dann mache ich einen eigenen Dienst daraus und jeder, der eben so einen Parameter lesen und schreiben möchte, der muß eben sich dieses Dienstes bedienen.

G: Das sind alles nur Softwaremodule. Das läuft auf diesem Prozessor, das ganze graue ist der Prozessor, der hier diese Software mit seinem Arbeitsspeicher und Programmspeicher drinnen hat. So kann man sich das vorstellen, ja?!

M: Das ist hier diese Linux. Der Linux-PC, quasi und da laufen diese ganzen Softwareprozesse. Also, wie auf dem normalen PC auch, dann könnte man sich vorstellen, da läuft ein Word und ein Excel und ein Outlook oder so ähnlich ist das hier auch. Das sind hier verschiedene Prozesse, die alle eine bestimmte Aufgabe haben und die miteinander kommunizieren.

G: In was für einer Sprache programmieren Sie das?

M: Das programmieren wir in C.

G: In C?

M: Ja. Vielleicht eine Sache vom Gedanken her, die auch noch neu ist. Wir möchten hier bestimmte Funktionen, die wir vielleicht jetzt nicht durch diese Funktionen, die hier implementiert sind, realisieren können, auf diesen Kommunikations-PC bringen. Also, wenn wir jetzt einen Fehler suchen und wir feststellen, mit dem was wir hier so drin implementiert haben, kriegen wir den Fehler nicht raus, dann muß man eine komplizierte Bedingung formulieren, bei der man dann Daten bekommt oder irgendwie so was, dann wollen wir das mit solchen Skripten machen. Das sind also kleine Programme, die man sich schnell schreiben kann in einer einfachen Sprache und die kann man dann auf diesem Kommunikations-PC laden und kann die Funktionalität, die hier zur Verfügung steht, noch einmal erweitern. Ich kann also dann noch komplexere Überwachungsfunktionen

wählen. Das ist die Bedeutung hinter diesem Kasten. Und da gibt es noch einen grünen Kasten, da steht „Webserver“ da, das ist jetzt nicht zu verwechseln mit dem, was wir für den Application-Server gesagt haben, das könnte so ein kleiner Webserver sein, so ein kleiner Miniwebserver auf diesem Kommunikations-PC sein und der könnte eine kleine Bedienoberfläche zur Verfügung stellen. Das heißt, wenn ich also jetzt kein BAUDIS habe und kein großes Antriebssystem, sondern ich habe nur so einen Regler mit so einem Kommunikations-PC, kann ich mich da dranstöpseln und kann die Parameter lesen und schreiben oder irgendwelche anderen.

G: Kann man auch weglassen den Webserver.

M: Könnte man auch weglassen.

G: Welche kann man nicht weglassen von diesen Blöcken hier?

M: Also, für unsere Funktion, damit die Anlage läuft, brauchen wir diese beiden.

G: Also, wir brauchen Request-Broker.

M: Einmal für Bedarfsdaten und bezüglich der Sollwerte.

G: Also, zwei Request-Broker, einmal für Parameterbedarfsdaten und einmal für bezüglich der Sollwerte. Dann brauchen wir noch mehr?

M: Dann brauchen wir das, was hier ist.

G: Die drei Blöcke noch weiter. Das ist also ein Request-Broker für Fehler und Events, für Parameterschnittstelle und das letzte ist für Softwaredownload. Diese fünf Broker sind unabdingbar.

M: Hm.

G: Also, die machen diesen Kommunikations-PC aus. Das sind die wesentlichen Funktionselemente.

M: Eigentlich ist nur der Webserver das einzige, was man weglassen könnte.

G: Ach, so.

M: Das gehört eigentlich alles zusammen.

G: Ja. Also letztlich ist nur der Webserver optional.

M: Ja.

G: Es geht halt darum, dass man vielleicht eine Erfindung aus diesem Kommunikations-PC definiert, dass man dann nur die notwendigsten Merkmale hat, damit man das also möglichst wenig umgehen kann. Deswegen frage ich da. Das heißt, wenn man hiervon was wegläßt, dann funktioniert das alles nicht mehr, von diesen fünf Brokern beispielsweise einen, dann ...

T: ... komplette Aufgabe, so wie er beschrieben hat, als Kommunikations-PC kann er dann nicht mehr erfüllen. Dann geht es halt nur in Teilbereichen.

G: Gut. Was haben wir noch für wesentliche Funktionskomponenten außer Kommunikations-PC?

M: Jawohl. Dann schauen wir noch einmal auf dieses Bild und kucken mal in diesen Application-Server.

G: BAUDIS-Application-Server kommt jetzt.

M: Hier sieht man jetzt also noch einmal diese Antriebsebene, die wir gerade gesehen haben, diese Kommunikations-PCs, das, was wir gerade angesehen haben, hier symbolisiert diese kleinen Request-Broker, die jeweils Informationen zur Verfügung stellen.

G: Jetzt sind es nur noch vier.

M: Jetzt sind es nur noch vier, richtig. Wenn ich, genau, eigentlich, eins, zwei, drei, vier. Weile diese beiden kommunizieren mit der Steuerung. Und die Informationen von und zur Steuerung, die haben wir ...

G: Ich sage es jetzt für das Band. Also, die beiden Request-Broker Parameterbedarfsdaten und zyklische Sollwerte sind dann auf der Seite 13 bei dem Kommunikations-PC weggelassen, weil diese nur mit der Steuerung kommunizieren.

M: Richtig. Hier diese Kommunikations-PCs bis zu 500 Stück, sage ich mal, und dann sieht man hier diesen Application-Server, schauen wir uns gleich näher an, und hier die verschiedenen Anwendungen. Hier sind mal zwei Anwendungen gezeigt. Diese ist die Hauptanwendung, hier sehen Sie diesen Webbrowser, also Internet Explorer oder Netscape, und hier quasi symbolisiert die Oberfläche, also die Webseiten, die dann zur Verfügung stehen für den User.

79

Jetzt schauen wir mal rein, was in diesem Application-Server drin ist: Auch wieder analog zu dem, was im Kommunikations-PC abläuft, eine modulare Softwarestruktur. Wir wollen ja in der Lage sein, das später mal zu erweitern oder einzelne Module auszutauschen. Sie sehen also hier vier Module gezeichnet, von denen es durchaus auch noch mehr geben könnte. Also, wenn ich noch neue Funktionen brauche, wenn ich z. B. einen neuen Reglertyp, wie z. B. den b maXX von BAUMÜLLER unterstützen möchte, brauche ich so ein neues Modul.

G: Ein neues Servermodul b maXX.

M: Zum Beispiel.

G: Anstelle von G-Drive.

M: Oder zusätzlich. Das sind also jetzt in diesen Modulen, da steckt jetzt die gesamte Funktion drin. Und um diese Funktionen, die wir hier implementieren nach außen zur Verfügung zu stellen, brauchen wir einen Webserver. Weil wir gesagt haben, wir wollen hier mit einem Webbrowser kommunizieren und Internetseiten zur Verfügung stellen, also brauchen wir einen Webserver. Das ist ein handelsübliches Ding, was man so hier verwendet. Weiterhin brauchen wir noch einen FTP-Server, damit können wir Dateien austauschen.

G: FTP heißt?

M: File Transfer Protocol. Und wir brauchen noch einen Telnet-Zugang. Das ist quasi eine Möglichkeit, um aus der Ferne Wartungen oder Einstellungen an diesem PC hier vorzunehmen. Also hier ist wieder nur die Software gezeigt, aber was darunter liegt, ist natürlich ein ausgewachsener Server-PC, also irgendein Internetprozessor, so wie sie hier so stehen. Was sind jetzt nun die einzelnen Aufgaben dieser Module? Hier dieses Servermodul G-Drive. Ich blätter mal um, weil da sehen wir jetzt noch einmal das Innere von diesem Application-Server. Ich denke, Sie können sich das jetzt vorstellen, das ist also quasi der innere Kasten. Alles, was da drin ist ...

G: Das ist BAUDIS-Server?

M: BAUDIS-Application-Server. Sieht man also hier noch einmal groß gezeichnet diese Servermodule hier und der Webserver und Telnet und FTP.

G: Hier sind es vier, hier sind es wieder fünf.

M: Hier sind es fünf, weil hier ist z. B. dieses Servermodul b maXX noch zusätzlich gezeichnet. Aber da wissen wir noch nicht genau, wie das aussieht. Deswegen ist das hier nur leer. Und bei manchen Sachen wissen wir noch nicht so ganz genau, was da alles rein kommt. Da sind wir eben jetzt gerade am Überlegen, aber diese Sachen sind schon relativ sicher. Servermodul G-Drive: Wie der Name schon sagt, das erledigt quasi alles, was mit dem G-Drive und der Kommunikation mit dem G-Drive zu tun hat. Alles was G-Drive spezifisch ist. Das Servermodul b maXX würde alles erledigen, was b maXX spezifisch ist. Hier ist diese Parameterschnittstelle zu nennen. Wir haben ja am Kommunikations-PC gesehen, es geht eigentlich nur um Parameter lesen und schreiben, hier diese Funktion Softwareupdate zu sehen, wir wollen ja irgendwie auch die Software für den Regler auf den Regler bringen. Bisher haben wir das immer so gemacht, dass wir das quasi von Hand dahintransportiert haben, also wir haben das auf so eine kleine Karte draufgespielt und die Karte eingesteckt und dann den Regler neu gebootet und dann war die Software drauf und jetzt wollen wir das so machen, dass es quasi auf Knopfdruck geht. Dass wir hier eine Funktion haben, und sagen: „drück drauf und dieser Regler wird mit einer neuen Software versehen“. Aber die Funktionen, die wir dafür brauchen, den Programmiercode, den man schreiben muß, der ist hier in diesem Servermodul drin. Ja und da sind noch so ein paar andere Funktionen. Die nennen sich Datensatzerstellung, Datensatzup- und download. Wir nennen diese Parameter oder bestimmte Gruppen von Parametern nennen wir Datensätze. Die sind für die Funktion des Reglers entscheidend und diese Datensätze können wir hier editieren und auf den Regler schicken oder wieder vom Regler laden. Und dann haben wir noch so eine Funktion die nennt sich Antriebsichern. Hier können wir quasi ein Image, also ein den Zustand des Reglers, Softwarezustand des Reglers abspeichern und daran können wir Änderungen vornehmen und wenn wir dann feststellen, das war wohl doch nichts, dann können wir diesen Zustand wieder zurückschrauben und das ist genauso wie vorher. Die Funktionen, die man dafür benötigt, sind da drin. Das ist G-Drive spezifisch, das ist b maXX spezifisch und alles, was man hier sieht, ist jetzt nicht mehr reglerspezifisch. Hier ist eine Servermoduldiagnose, hier tauchen diese Fehler und Alarme wieder auf, von denen ich beim Kommunikations-PC gesprochen habe, wir hatten ja diesen Request-Broker Fehler und Events, der Fehler zur Verfügung stellt, wenn also ein Fehler auftritt, dann informiert er mich, und das ist quasi das Gegenstück dazu. Also der nimmt die Fehler entgegen bzw. der konfiguriert den Broker auf dem Kommunikations-PC und sagt ihm, auf welche Fehler er reagieren soll oder auf welche Ereignisse er reagieren soll. Hier ist eine Langzeitaufzeichnung gezeigt, wir wollen z. B. für Analysezwecke bestimmte Parameter über viele Stunden oder viele Tage

aufzeichnen und die dann hinterher graphisch anschauen und das ist dann hier noch, und dann können sicherlich noch ein paar Sachen dazukommen. Das Servermodul Anlagenkonfiguration: Wir müssen ja, wenn man jetzt an den Aufbau und die Inbetriebnahme einer solchen Anlage denkt, muß man ja erst mal die Konfiguration der Anlage irgendwie in dieses System hineinbekommen. Also Konfiguration, damit meine ich jetzt: Da ist jetzt ein Druckturm und da ist noch ein Druckturm und das ist noch ein Druckturm und da ist ein Falz und da habe ich jetzt an jedem Druckturm so und so viele Antriebe. Das muß ich ja irgendwo mal einrichten, damit ich dann mit diesem System arbeiten kann und das findet in diesem Modul statt. Da haben wir also ein Anlagenbild, ich weiß nicht, ob Sie das BAUDIS schon mal in Echt gesehen haben, da gibt es also ein Anlagenbild, da werden alle Antriebe dargestellt, ich kann Ihnen da mal kurz einen Ausschnitt davon zeigen, wie man sich das vorstellen kann. Wenn Sie noch einmal diesen Ausschnitt betrachten hier, dann sieht man hier so ein Anlagenbild, das ist also eine symbolische Darstellung. Hier sieht man so verschiedene Druckwerke und hier sieht man Motoren und immer wenn da ein Fehler auftritt, da fängt dieser Motor an zu blinken. Und da kann also der Drucker oder der Bediener hat da den Überblick über seine Anlage und sieht sofort, hier ist ein Fehler aufgetreten, hier ist irgend was passiert, hier muß ich darauf reagieren. Und um dieses Anlagenbild zu erstellen, und die notwendigen Adressen und notwendige Konfiguration hier ...

G: Aber das paßt nicht zusammen.

M: Ach, Gott. Um das vorzuhalten, dafür brauche ich dieses Servermodul. Ich denke, wir haben es auch eigentlich gleich. Eine Minute brauche ich noch.

JDBC = Java Data Base
Connection

DB in MByte - Beeil

für jeden Modul Beeil

hier: Ethernet zur Diagnose?

— Kommunikation - PC
= embedded PC

● bisher B AUD, logisch
zu-
mal strukturiert, jetzt
dezentral, Aufgaben
an einzelne kommunika-
tions - PC delegieren

keine neue HW

keine neue Techno-
logie

Erfindung liegt in
Anwendung einer de-
zentralen Diagnose-
Struktur (Kommunikation
- PCs)

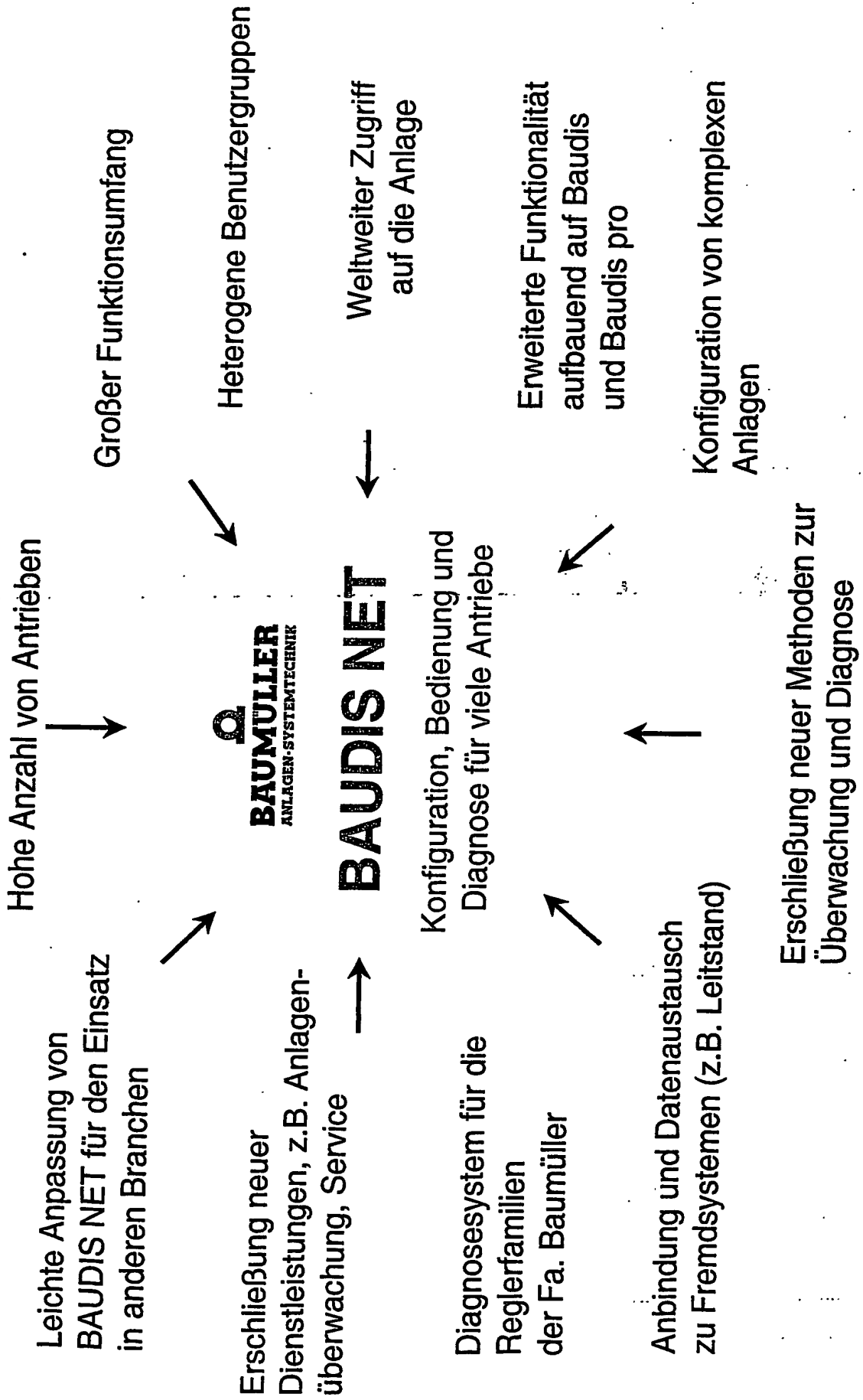
Vorsorgende Fehlerdiag-
nose, Veränderungen erfassen
statistische Prozeßkontrolle

Dezentrales Diagnosesystem
zur Diagnose u. Inbetrieb-
nahme von elektrischen
Antriebsystemen

PATENTANSPRUCH

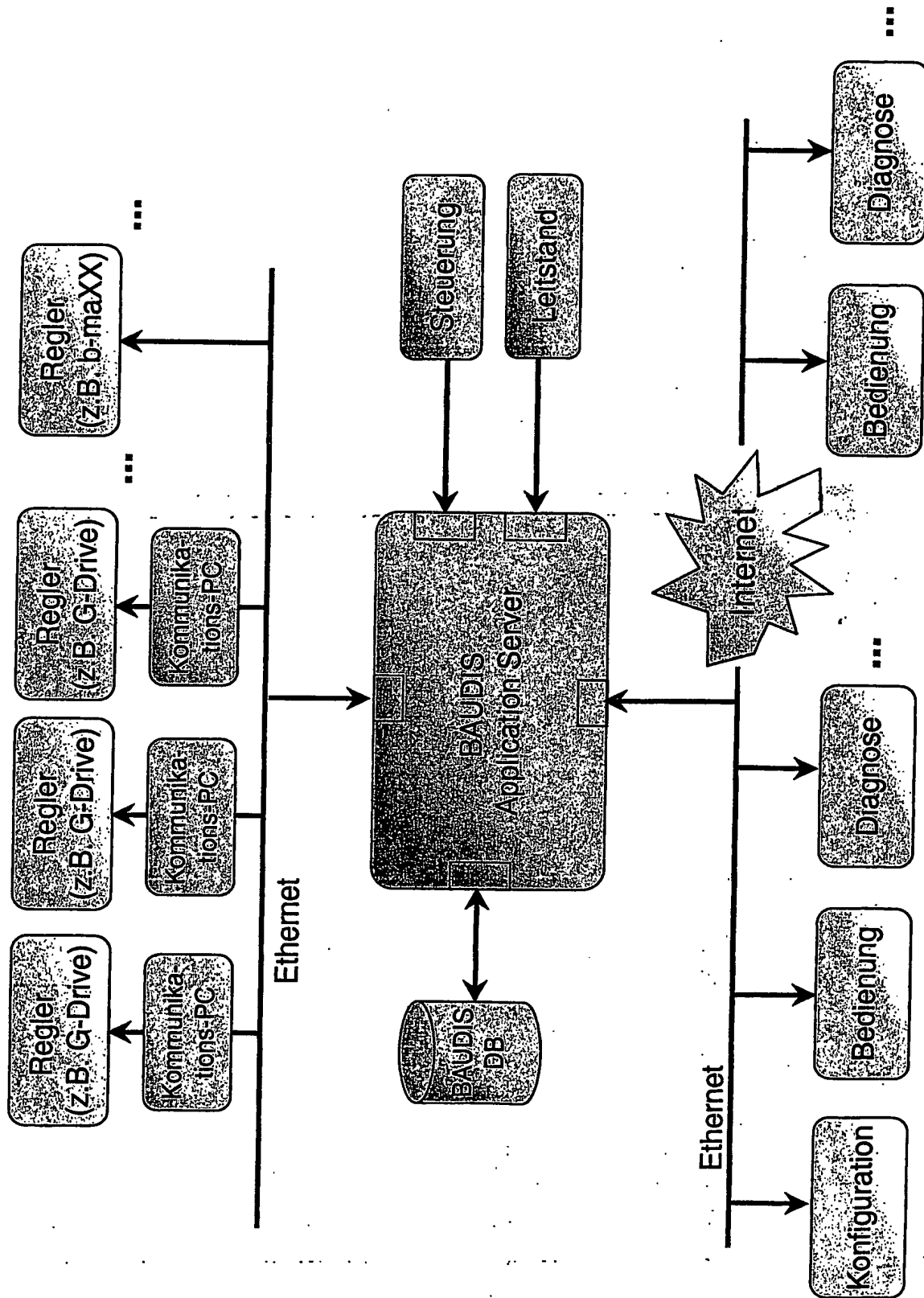
5 Anordnung und Verfahren zur Fehler-Diagnose und/oder -Analyse eines
oder mehrerer technisch-physikalischer Prozesse, insbesondere elektri-
scher Antriebsvorgänge, die unter der Steuerung, Regelung und/oder
Überwachung durch einen oder mehrere Prozeßrechnerknoten ablaufen,
wobei über wenigstens einen gemeinsamen Bus die Prozeßrechnerknoten
mit wenigstens einem Diagnoserechnerknoten verbunden sind, in dem eine
10 oder mehrere Diagnosedienste und/oder -funktionen implementiert sind, die
dem oder den Prozessen und/oder dem oder den Prozeßrechnerknoten
und/oder den darin ablaufenden Verarbeitungsprozessen zugeordnet sind,
gekennzeichnet durch den gleichzeitigen Einsatz asynchroner und syn-
chroner Kommunikationssysteme auf Leit- und Steuerungsebene, wobei die
15 Sollwertvorgabe asynchron von der Leitebene erfolgt.

BAUDIS NET - Anforderungen



215

Entwurf: BAUDIS NET Grobstruktur

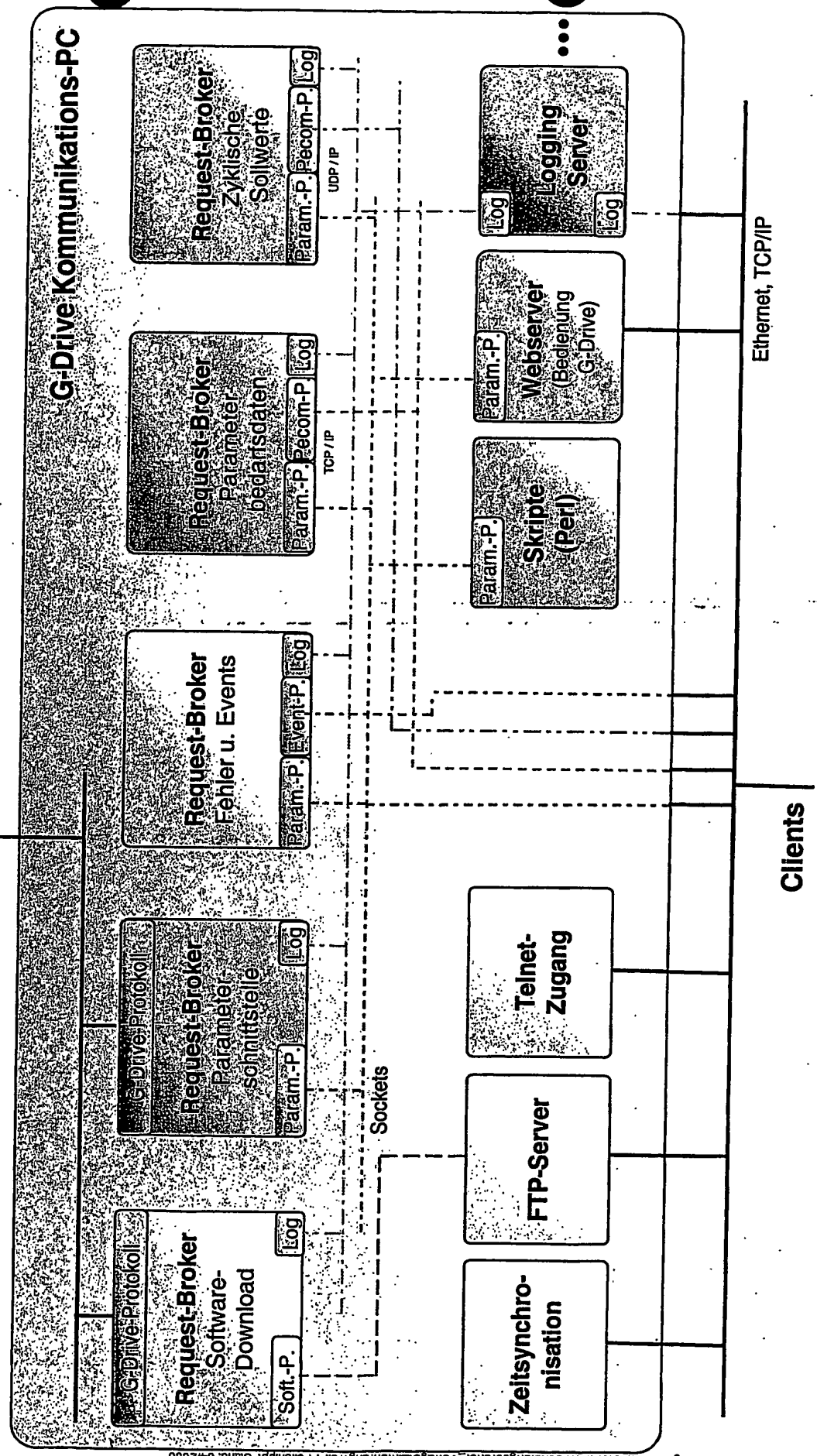


3/5

Kommunikations-PC - Struktur

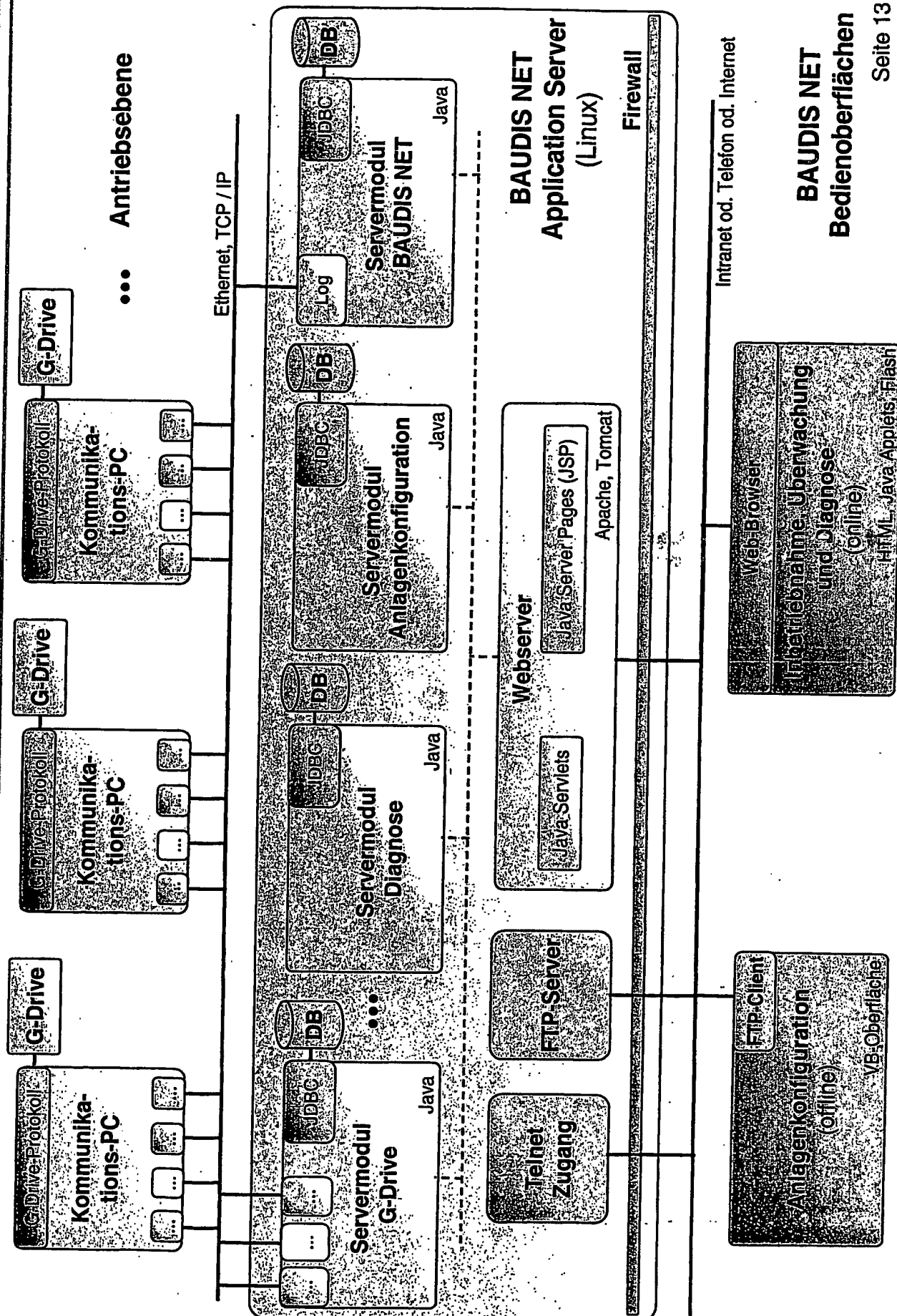
Serielle synchrone
Schnittstelle,
TCP/IP

G-Drive



4/5

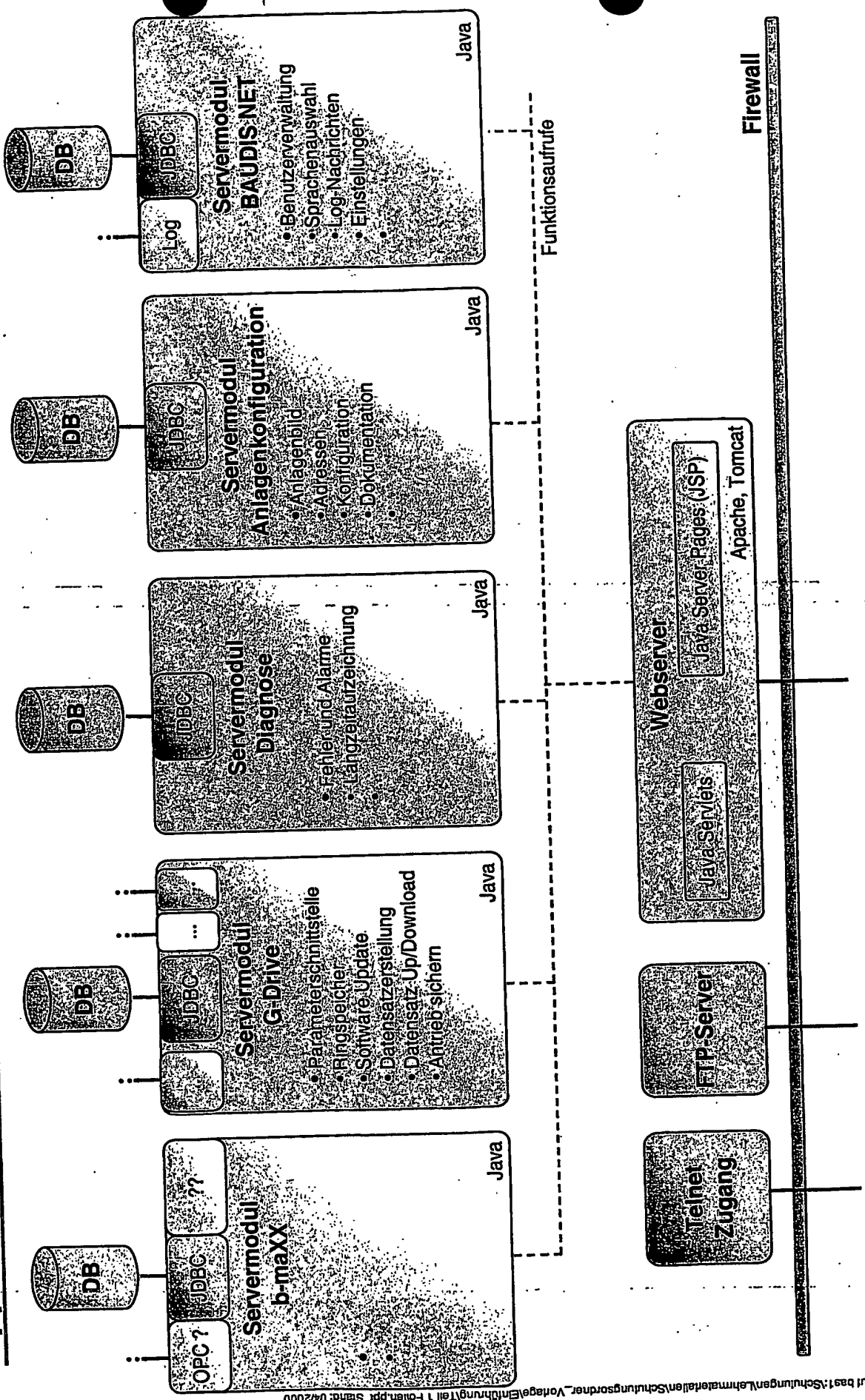
BAUDIS NET - Struktur



5/5

Spezifikation BAUDIS NET

Application Server - Struktur



This Page is inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☒ BLACK BORDERS
- ☒ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☒ FADED TEXT OR DRAWING
- ☐ BLURED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☒ COLORED OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REPERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

**As rescanning documents *will not* correct images
problems checked, please do not report the
problems to the IFW Image Problem Mailbox**